Introduction to AppleScripting Microsoft Entourage

MacTech Magazine • May • 2007

MACTECH®

The Journal of Macintosh Technology

EFI

Apple's Transition From Open Firmware to Extensible Firmware Interface

Also In This Issue:

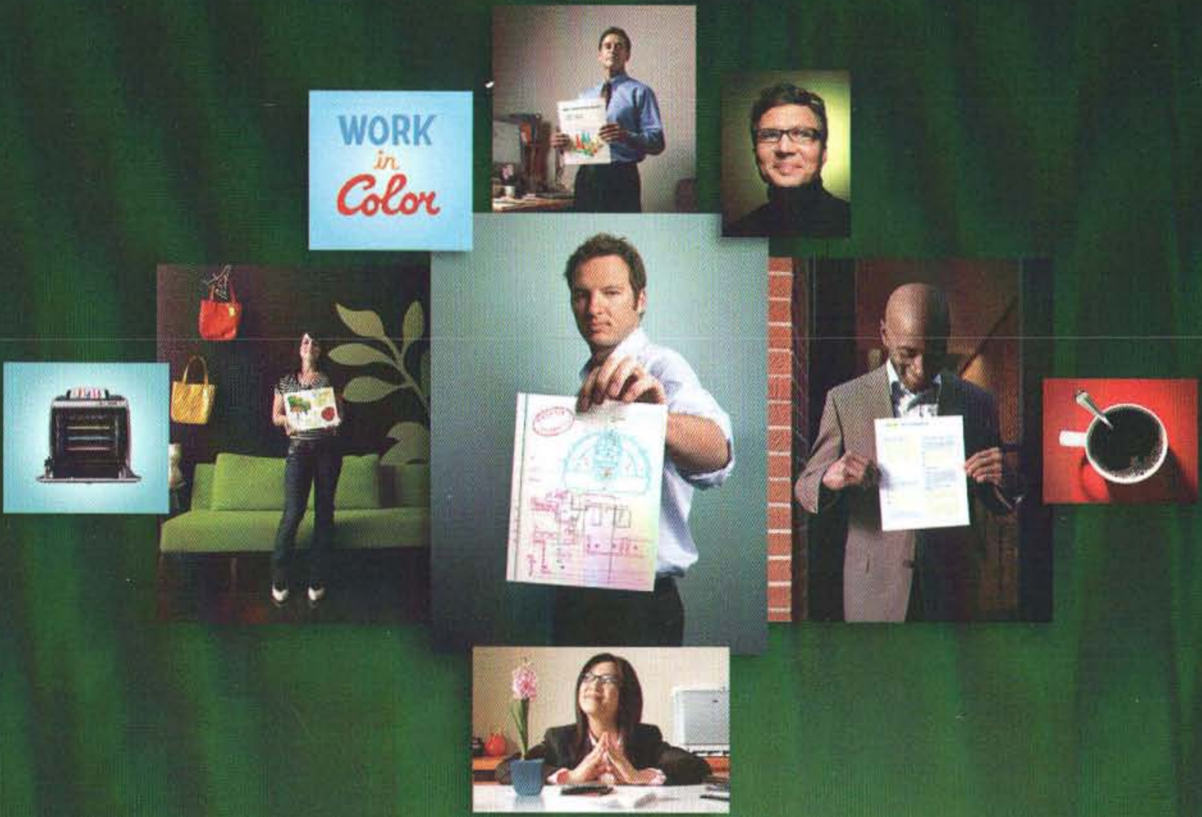
Perl 6 On XCode

Learn F-Script in 20 Minutes

MACTECH.COM

$8.95 US, $12.95 Canada

ISSN 1067-8360 Printed in U.S.A.



IT'S NOT EXPENSIVE TO LOOK EXPENSIVE.
HP COLOR LASERJETS STARTING AT $399.

With prices this low, it makes more sense than ever to get an HP Color LaserJet printer. Enjoy HP's renowned print quality when teamed with HP ColorSphere toner. HP has been *PC Magazine*'s Readers' Choice for service and reliability 15 years in a row. HP LaserJets, the reliable workhorse printers for business.

FASTEST
HP Color LaserJet 3800n.
Up to 22 pages a minute.
There's no deadline it can't meet.
$899.*

FASTER
HP Color LaserJet 3600n.
Up to 17 pages a minute.
A great fit for a growing business.
$599.*

FAST
HP Color LaserJet 2600n.
Up to 8 pages a minute.
More affordable than ever.
$399.

To learn more or see special offers, go to hp.com/go/mac.
Call 1-800-888-3119. Visit your reseller or retailer.

*After $100 instant savings. ©2007 Hewlett-Packard Development Company, L.P.
Estimated U.S. retail price. Actual price may vary. Simulated images.

He's busy doing a software update on 1,037 computers.

Hard to tell though, isn't it? That's because he's automating it using FileWave® software. Totally cross-platform, totally automated. So, are you an IT administrator? Tired of manually installing OS updates, security updates and patches? Or maybe you just aren't keeping your software up to date? Automate it with FileWave® and save time for the important things.

Hundreds of satisfied customers agree: FileWave® just works.

FILEWAVE®

www.filewave.com

"We can do all updates via FileWave® with 99.9% success (you wouldn't believe me if I had said 100%)"

– David Zubradt, FileWave Administrator for a leading US cable television corporation.

"I was skeptical of the FileWave® product because it seemed too easy and too good to be true. After piloting the product for two months on PC's and Mac's, we ordered 4,400 clients because it worked that well. I love having the ability to manage software on both platforms from one console. FileWave® allows our technology staff to focus on other responsibilities while it deploys, repairs, and removes software. FileWave® is the single best purchase with the most "bang for the buck" I have seen in years. I couldn't be more pleased with the product."

– Mike Kneebone, North Central High School, Indianapolis"

MAXIMUM
PERFORMANCE
DATA STORAGE **TAPE**

For over three decades, Maxell has led the Data Storage media industry with advanced technologies and progressive products. Today, Maxell continues to lead the evolution of Computer Data media. Utilizing innovative, proprietary NeoSMART™ technologies, Maxell tapes give you complete confidence that your data has been stored on some of the most stable, durable and reliable media available. Maxell tapes also help you meet stringent government policies, including Sarbanes Oxley and HIPAA. Maxell will continue taking tape to the terabyte level and beyond.

Maxell. The tape to trust.

LTO

DLT S4

DLT

DDS DAT

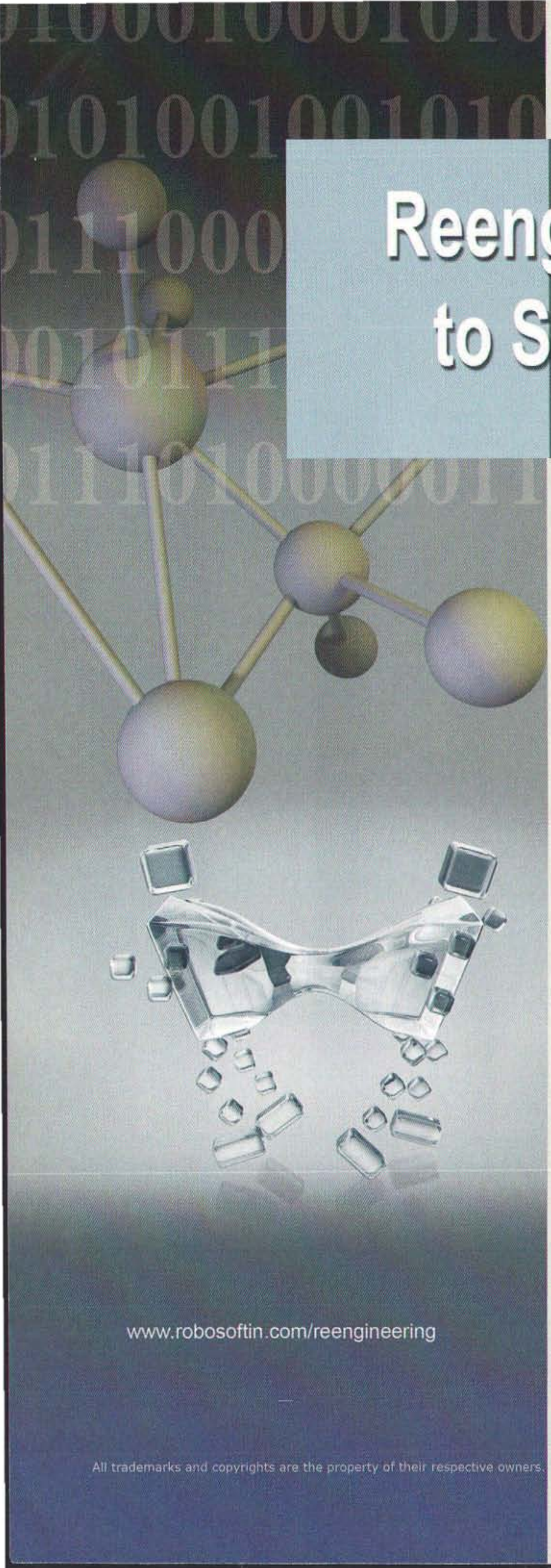
3590E

maxell
www.maxell.com

DLT ULTRIUM LTO Digital Data Storage NeoSMART

TABLE OF CONTENTS

ARTICLES & DEPARTMENTS



Reengineering to Succeed!

Are you looking for new ways to improve mind and market share?

Competition is forcing companies to expand their sphere of influence, striving for the new markets. Existing software systems need to move with the times too. Today, most organizations are aiming to shift from the existing legacy systems to newer technologies in order to respond to market needs mandated by rapidly changing technologies and delivery platforms. Wouldn't you be interested in breathing new life into your existing applications and leverage the latest technologies that companies like Apple and Microsoft are releasing with their new OS?

Robosoft provides cost effective application migration services to leverage the advantage of latest Mac technologies in your applications and certifying them for Universal Binary Compliance. Robosoft also assists in enhancing your existing Windows applications Vista savvy. Robosoft has a specific advantage due to its strong partner relationship with world technology leaders thus helping customers to leverage on faster access to new technology innovation, faster turn around time on technology related issues.

Partner with us and reengineer your way to Success!

www.robosoftin.com/reengineering

Mac Universal

Robosoft®

Communicate With Us

Department E-Mails

Orders, Circulation, & Customer Service
cust_service@mactech.com

Press Releases
press_releases@mactech.com

Ad Sales
adsales@mactech.com

Editorial
editorial@mactech.com
(Authors only, no pr)

Accounting
accounting@mactech.com

Marketing
marketing@mactech.com

General
info@mactech.com

Web Site
http://www.mactech.com

In this electronic age, the art of communication has become both easier and more complicated. Is it any surprise that we prefer **e-mail?**

If you have any questions, feel free to call us at 805/494-9797 or fax us at 805/494-9798.

If you would like a subscription or need customer service, feel free to contact MacTech Magazine Customer Service at 877-MACTECH

We love to hear from you! Please feel free to contact us with any suggestions or questions at any time.

Write to letters@mactech.com or editorial@mactech.com as appropriate.

MACTECH®

The Journal of Macintosh Technology

A publication of **XPLAIN** CORPORATION

The Editorial Staff

Publisher & Editor-in-Chief: Neil Ticktin
Executive Editor: Edward R. Marczak
Editor-at-Large: Dave Mark
Business Editor: Andrea Sniderman
Editor-at-Large, Open Source: Dean Shavit
Managing Editor: Dennis Bower
Production: David Allen
Staff Writer: Jeffrey A. Rochin
Staff Writer: Marianne Shilpa Jacobie

Xplain Corporation Senior Staff

Chief Executive Officer: Neil Ticktin
President: Andrea J. Sniderman
Accounting: Marcie Moriarty
Customer Relations: Susan Pomrantz
Board of Advisors: Steven Geller, Alan Carsrud

Regular Columnists

QuickTime ToolKit: by Tim Monroe
Patch Panel: by John C. Welch
The Source Hound: by Dean Shavit
Reviews/KoolTools: by Michael R. Harvey
AppleScript Essentials: by Ben Waldie
Mac In The Shell: by Ed Marczak

Board of Advisors

Chairman: Dave Mark,
Jordan Dea-Mattson, Steven Geller, Bruce Friedman, and Richard Kimes

Contributing Editors

Michael Brian Bentley, Gordon Garb, Vicki Brown, Chris Kilbourn
Marshall Clow, Rich Morin, Will Porter, Tom Djajadiningrat, Avi Rappoport,
Andrew S. Downs, Cal Simone, Steve Sisak

MacTech Magazine (ISSN: 1067-8360 / USPS: 010-227) is published monthly by Xplain Corporation, 850-P Hampshire Road, Westlake Village, CA 91361-2800. Voice: 805/494-9797, FAX: 805/494-9798. Domestic subscription rates are $47.00 per year. Canadian subscriptions are $59.00 per year. All other international subscriptions are $97.00 per year. Domestic source code disk subscriptions are $77 per year. All international disk subscriptions are $97.00 a year. Please remit in U.S. funds only. Periodical postage is paid at Thousand Oaks, CA and at additional mailing office.

POSTMASTER: Send address changes to **MacTech Magazine**, P.O. Box 5200, Westlake Village, CA 91359-5200.

From the Editor

May is a fine time to catch up on spring cleaning, get outside a bit more (well, for those of us in the Northern Hemisphere) and to finish off projects that you've put off for too long. Like OS 10.5. The Leopard delay has been written about extensively at this point. Frankly, we're all thankful for this at MacTech. We'd rather see a better product go gold than one that got rushed out the door. October will be here before you know it, and MacTech will be covering Leopard when it ships.

Back in the present, we have enough articles to keep you in the know. I'm very happy to have an article introducing **FScript**, written by the author of FScript himself! Created by **Philippe Mougin**, FScript is a smalltalk-inspired scripting language that lets you poke around the bowels of your Mac, prodding live data structures and giving you nearly unrivaled control. Some commercial applications, such as Colloquy and Daylite, have even adopted FScript as a plug-in or internal scripting language.

José Cruz presents us with two (TWO!) articles this month. First, he shows us how to integrate **perl 6** into XCode. Perl is such a useful scripting language, perhaps getting a little overshadowed by some of the newer languages, such as Python and Ruby, but it's still ingrained in many people's minds (and fingers). Of course, version 6 adds even more to the solid foundation that it already has. Switching scripting languages, José shows us a way to create an **uninstaller using AppleScript** and AppleScript studio. He brings solid reasons why you may want to do this, especially if your installer may not be built with Apple's packaging tools... or even if it is.

Check out **Mike Harvey's** review of **3Ware's Sidecar** external SATA disk. He lays out why this is such a nice piece of gear. Having rolled this out at a client or two, it's a perfect fit between "Firewire isn't good enough" and " an Xserve RAID is too expensive".

Criss Myers returns with another article that hits the lower-level components of the system we use every day. As we work through the Intel transition, another major component has been replaced: OpenFirmware is no more, being replaced by **EFI**. Criss guides us to what's new, and what you need to know.

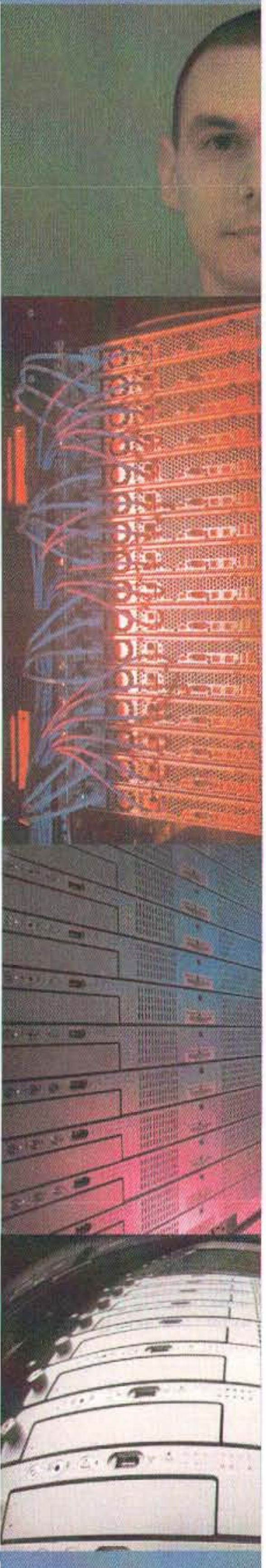
Last, but not least, we give our best wishes to author **Ben Waldie** who runs his last article in his column this month. However, it's not goodbye! Ben will still be doing what he does best: automate workflows using AppleScript. He'll hopefully have enough time to grace us with some more words of wisdom from time to time. He will continue to be found through his website at http://www.automatedworkflows.com. Thanks for many, many excellent articles, Ben!

Find this and more, including this month's MacTech Spotlight on **Paul Kafasis** from **Rogue Amoeba**, inside!

Edward Marczak
Executive Editor

This issue is dedicated to everyone at Virginia Tech.

As we were putting the finishing touches on this issue, we learned of the recent tragedy at the school. With an institution that's so steeped in technology, particularly one that put together one of the first massive Macintosh based compute clusters, it's easy for us to think of the technology first. MacTech even had an issue featuring the Virginia Tech Cluster on its cover. However, this is clearly about the people involved, and everyone touched by this event. Our hearts and minds are with them.



Hard Drive Upgrade
15" PB G4 Logic Board Repair
PowerBook G3 Fan Replacement
iPod Mini Hard Drive Repair
G4 Mercury Keyboard Replacement
iPod Display Repair
14" iBook G3 Modem Repair
iPod Battery Replacement

ifixit

Mac + iPod Parts & Upgrades

You fix it. We make it **easy**.

We enable you to repair your Apple PowerBook, iBook, MacBook or iPod yourself.

With **thousands** of **Mac + iPod parts**, we can supply you the exact Apple part that you need. We sell the tools that repair techs use and provide **free Fixit Guides** so you can take it apart yourself. Go ahead and take a look. Each Fixit Guide has detailed disassembly instructions that walk you through the process of accessing and replacing components in your machine. The Fixit Guides are available for **free on our website.**

We sell parts for Mac laptops and iPods. We have great deals on:

- **AC Adapters**
- **Batteries**
- **Case Components**
- **DC & Sound Cards**
- **Displays**
- **Fans**
- **Hard Drives**
- **Keyboards**
- **Logic Boards**
- **Modems**
- **Optical Drives**
- **Processors**
- **RAM**
- **Wireless Cards**

Broke your Mac or iPod, but don't feel comfortable diving in without any experience? What Apple doesn't tell you is that you can fix your Apple laptop or iPod yourself. All you need is the part, the right tools, and a way into your Apple product. That's where we come in.

Our goal is to empower people to do repairs themselves by giving them the knowlege, tools, and parts to do so. So far, we've helped over 20,000 people repair their computers and iPods. We'd be happy to help you, too. Check us out online or give us a call at **1-866-61-FIXIT** (34948).

Use this coupon for $5 off your next purchase at www.iFixit.com

[MACTECH043]

by Ben Waldie

Introduction to AppleScripting Microsoft Entourage

For some time now, we have been discussing scripting the Microsoft Office applications, partially in preparation for the forthcoming release of Office 2008, in which Microsoft has announced that Visual Basic macros will no longer be supported. So far, we have explored the AppleScript support in Microsoft Word, Excel, and PowerPoint, three applications that currently (in Office 2004) include Visual Basic macro support. This month, we are going to discuss the fourth major Office application, Entourage, a popular email and project management client.

Unlike the other Office applications, Entourage does not include Visual Basic macro support. So, when Office 2008 is released, there won't be any macros requiring conversion. However, Entourage does possess fairly comprehensive AppleScript support. Using AppleScript, it is possible to manipulate various types of elements in Entourage, including messages, contacts, events, tasks, notes, and more. In this month's column, we will explore ways of manipulating some of these elements. Let's get started.

Working with Messages

Probably the primary elements with which you will want to interact are messages. Using AppleScript, it is possible to create messages, read messages, send messages, move messages from one folder to another, and more.

In Entourage, there are two primary types of messages, incoming and outgoing. While each of these types possesses its own set of properties, both types inherit a number of common properties from the more generic **message** class.

Creating an Outgoing Message

To create a new outgoing message, use the **make** command, followed by the class **outgoing message**, and optionally, properties to be assigned to the message. For example:

```
set theSubject to "Some Subject"
set theBody to "Some Body"
tell application "Microsoft Entourage"
    make new outgoing message with properties
{subject:theSubject, content:theBody}
end tell
--> outgoing message id 11285 of application "Microsoft
Entourage"
```

Take note that, when creating a new outgoing message, it will not automatically be visible in Entourage's interface. Regardless, it is created. To make a newly created outgoing message visible in Entourage's interface, the message must be opened. Fortunately, the **make** command will return as its result a reference to the newly created message. This reference may be placed into a variable, which can then be opened using the **open** command, as demonstrated below.

```
set theSubject to "Some Subject"
set theBody to "Some Body"
tell application "Microsoft Entourage"
    set theMessage to make new outgoing message with
properties {subject:theSubject, content:theBody}
    open theMessage
end tell
```

Adding Recipients to a Message

In the examples that we have discussed above, we have specified certain properties for the outgoing message, which are assigned when the message is created. Specifically, we have specified a subject and a body for the message. However, an outgoing message cannot be sent without first assigning it recipients.

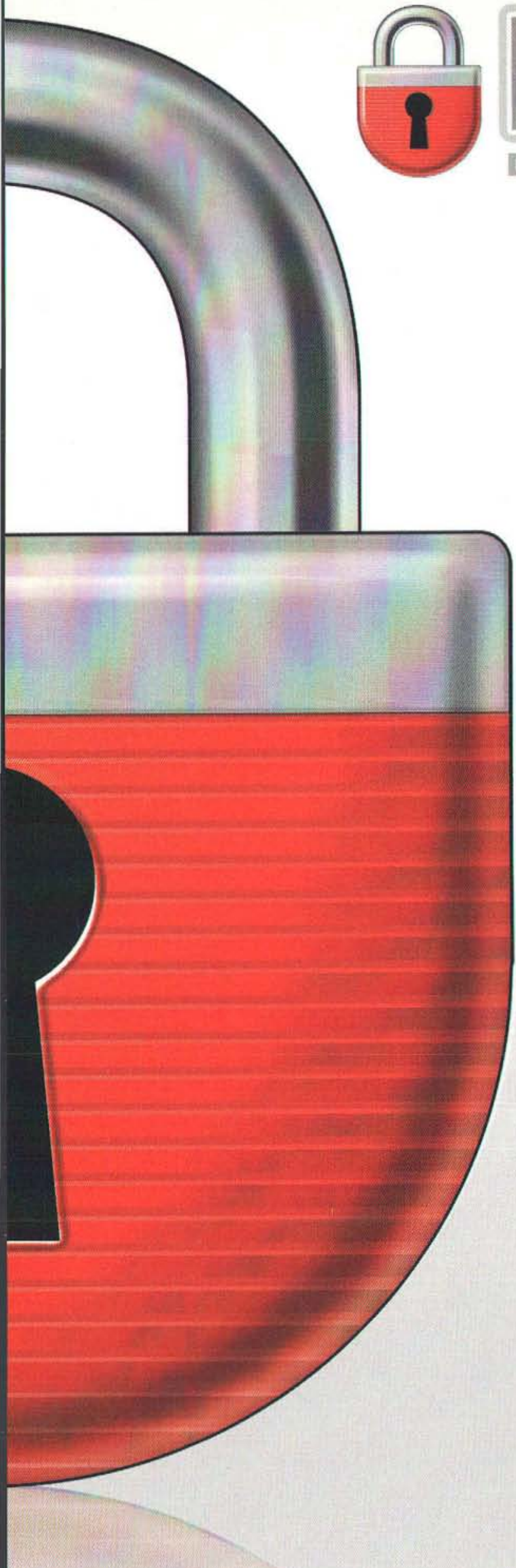
When assigning recipients to a message, each recipient must consist of two bits of information. The first bit of information is the address, and the second is the type of recipient (to, cc, bcc). The address of a recipient is broken down even further, into two more parts - an email address and a display name (i.e. first and last name of the recipient).

To accommodate this, Entourage's AppleScript support includes a **recipient** class, which possesses an **address** property and a **recipient type** property. The value of a recipient's **address** property translates to the **address** class in Entourage, which has a **display name** property and an **address** property of its own. Put together in AppleScript code, this all translates to the following for each recipient in the list of recipients:

```
{address:{display name:"RecipientName",
address:"RecipientEmailAddress"}, recipient type:to
recipient}
```

Now, let's put this together to assign recipients as a message is created. The following example code will create an outgoing message in Entourage, addressing it to **Ben Waldie**, and copying **MacTech Editorial**, the result of which is shown in figure 1.

```
set theSubject to "Some Subject"
set theBody to "Some Body"
tell application "Microsoft Entourage"
    set theRecipients to {{address:{display name:"Ben
Waldie", address:"ben@automatedworkflows.com"}, recipient
type:to recipient}, {address:{display name:"MacTech
Editorial", address:"editorial@mactech.com"}, recipient
type:cc recipient}}
    set theMessage to make new outgoing message with
properties {recipient:theRecipients, subject:theSubject,
content:theBody}
    open theMessage
end tell
```



MacScan

DETECTS, ISOLATES & REMOVES SPYWARE

MAC SPYWARE PROTECTION

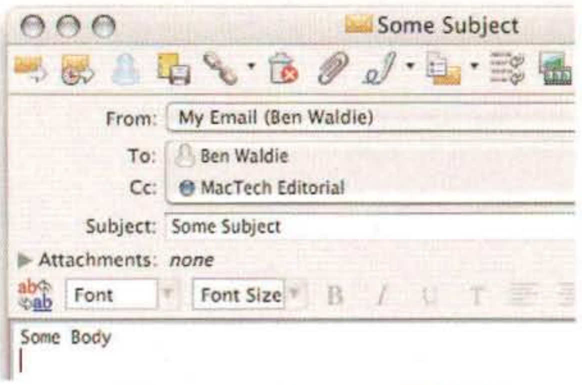
Protect your security and privacy with MacScan, the premier Anti-Spyware program for Mac OS X

Spyware exists--and it's a growing threat to the online community. As the number of Mac users increases, so do the threats to your personal information. MacScan delivers peace of mind by auditing your computer for spyware and cleaning up clutter from casual Internet browsing to protect privacy and security. MacScan's Cookie Blacklist allows you to safely scan and remove known tracking cookies from popular web browsers.

- Automatically updates spyware definitions to guard against the latest threats
- Supports 9 browsers including Firefox, Safari, and Netscape
- Over 8000 blacklisted cookies in the database

Start Protecting Yourself From Spyware
DOWNLOAD YOUR FREE TRIAL TODAY
http://macscan.securemac.com

Mac Universal



From: My Email (Ben Waldie)
To: Ben Waldie
Cc: MacTech Editorial
Subject: Some Subject
Attachments: *none*

Some Body

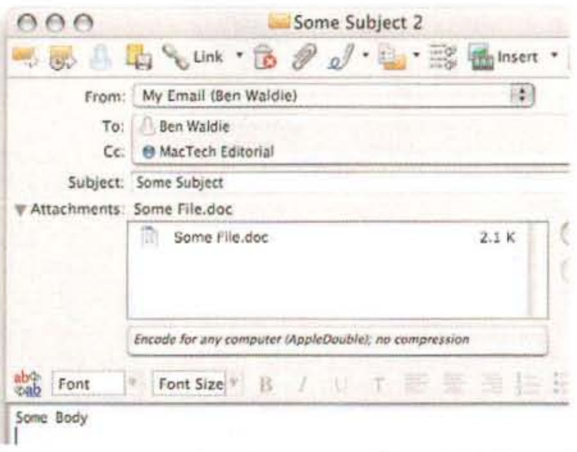
Figure 1. An Outgoing Message with Recipients

Adding Attachments to an Outgoing Message

Another common task when creating outgoing messages is adding attachments. One or more files may be added to an outgoing message as attachments by providing the paths to those files when the message is created, as follows:

```
set theSubject to "Some Subject"
set theBody to "Some Body"
set theAttachment to choose file with prompt "Select a file
to attach:" without invisibles
tell application "Microsoft Entourage"
     set theRecipients to {{address:{display name:"Ben
Waldie", address:"ben@automatedworkflows.com"}, recipient
type:to recipient}, {address:{display name:"MacTech
Editorial", address:"editorial@mactech.com"}, recipient
type:cc recipient}}
     set theMessage to make new outgoing message with
properties {recipient:theRecipients, subject:theSubject,
content:theBody, attachments:theAttachment}
     open theMessage
end tell
```

In this case, we have only added a single attachment to an outgoing message. Adding multiple attachments is done in the same manner. Just specify a list of paths, rather than a single path. Figure 2 shows the result of the previous example code.



From: My Email (Ben Waldie)
To: Ben Waldie
Cc: MacTech Editorial
Subject: Some Subject
Attachments: Some File.doc
Some File.doc 2.1 K
Encode for any computer (AppleDouble); no compression

Some Body

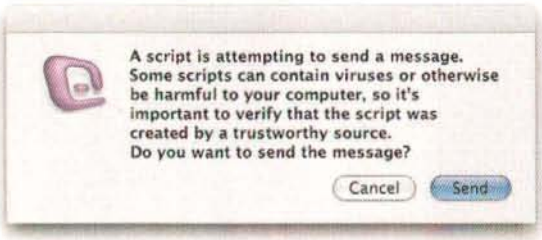
Figure 2. An Outgoing Message with an Attachment

Sending an Outgoing Message

Once you have created an outgoing message, you will most likely want to send it. Earlier, we discussed opening outgoing messages, since they are not visible by default. Well, if you are going to immediately send an outgoing message after creating it, then you might not need to open it. Regardless, to send a message, use the send command, followed by a list of the messages you want to send. For example:

```
send theMessage
```

Please note that, when first sending a message via AppleScript, Entourage will display a dialog notifying the user that a script is attempting to send a message (figure 3). The user must click a *Send* button before the script can proceed. Keep this in mind, and be prepared for this occurrence.



**A script is attempting to send a message.
Some scripts can contain viruses or otherwise be harmful to your computer, so it's important to verify that the script was created by a trustworthy source.
Do you want to send the message?**

Figure 3. Entourage's Send Message Warning Dialog

Working with Incoming Messages

So far, we have been discussing ways of working with outgoing messages. Incoming messages possess most of the same properties that outgoing messages possess, so interacting with them is very similar.

To retrieve a list of currently selected incoming messages, reference the `selection` property of the application. For example:

```
tell application "Microsoft Entourage"
     selection
end tell
--> {incoming message id 136 of application "Microsoft
Entourage"}
```

Once you have a reference to an incoming message, you can retrieve any of its properties. For example, the following code will retrieve the subject of a selected incoming message.

```
tell application "Microsoft Entourage"
     set theSelection to selection
     set theCurrentMessage to item 1 of theSelection
     subject of theCurrentMessage
end tell
--> "RE: Meeting Today!"
```

Forwarding a Message

To forward a message to another recipient, you can use Entourage's forward command. This command requires a reference to a message to be forwarded as its direct parameter.

Optional parameters include a to recipient for the forwarded message, and whether the forwarded message's window should be opened in Entourage's interface. For example:

```
tell application "Microsoft Entourage"
    set theSelection to selection
    set theCurrentMessage to item 1 of theSelection
    forward theCurrentMessage to "ben@automatedworkflows.com"
with opening window
end tell
--> window "FW: Meeting Today!" of application "Microsoft
Entourage"
```

Take note that the result of the **forward** command in the example code above is a reference to the window of the forwarded message. This is because we specified that the forwarded message should be opened. If we had specified for the message not to be opened, then an outgoing message reference would have been returned instead. For example:

```
tell application "Microsoft Entourage"
    set theSelection to selection
    set theCurrentMessage to item 1 of theSelection
    forward theCurrentMessage to "ben@automatedworkflows.com"
without opening window
end tell
--> outgoing message id 11277 of application "Microsoft
Entourage"
```

Moving a Message to a Folder

You may have a need to move incoming messages from one folder to another via AppleScript. To do this, you can use the **move** command. When using this command, you must provide a reference to the message you want to move, along with a reference to the folder into which you would like the message to be moved. For example, the following code would move the currently selected message into a folder called "Some Folder".

```
tell application "Microsoft Entourage"
    set theSelection to selection
    set theCurrentMessage to item 1 of theSelection
    move theCurrentMessage to folder "Some Folder"
end tell
```

Keep in mind that since folders have a hierarchy in Entourage, then they must be addressed within this hierarchy via AppleScript. The following example code would move a message into a folder named "Another Folder", which resides inside of a folder named "Some Folder".

```
tell application "Microsoft Entourage"
    set theSelection to selection
    set theCurrentMessage to item 1 of theSelection
    move theCurrentMessage to folder "Another Folder" of
folder "Some Folder"
end tell
```

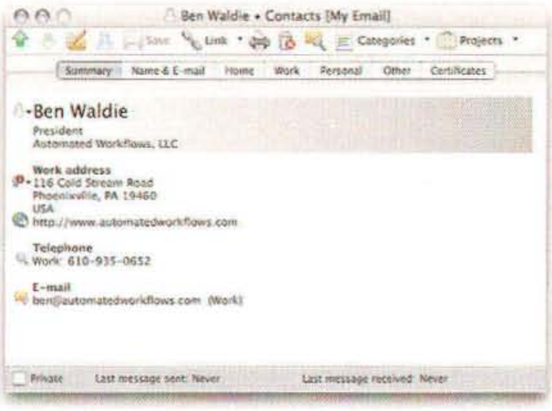
Working with Contacts

Now that we have discussed ways of interacting with messages in Entourage, let's discuss ways of interacting with other scriptable elements. First, we will take a look at contacts.

To create a new contact in Entourage, use the **make** command, much in the same way that we discussed creating outgoing messages. For the sake of efficiency, you will probably want to apply properties to the contact as it is created. To do this, use the **make** command's optional **with properties** parameter.

The following example code demonstrates how this is done. This particular code will create a contact, complete with a first name, last name, company, job title, business address, business web page address, and email address. Figure 4 shows an example of the newly created contact.

```
tell application "Microsoft Entourage"
    make new contact with properties {first name:"Ben", last
name:"Waldie", company:"Automated Workflows, LLC", job
title:"President", business phone number:"610-935-0652",
business address:{street address:"116 Cold Stream Road",
city:"Phoenixville", state:"PA", zip:"19460",
country:"USA"}, business web
page:"http://www.automatedworkflows.com", email
address:{{label:"Work",
contents:"ben@automatedworkflows.com"}}}
end tell
--> contact id 18 of application "Microsoft Entourage"
```



Ben Waldie
President
Automated Workflows, LLC

Work address
116 Cold Stream Road
Phoenixville, PA 19460
USA
http://www.automatedworkflows.com

Telephone
Work: 610-935-0652

E-mail
ben@automatedworkflows.com (Work)

Private   Last message sent: Never   Last message received: Never

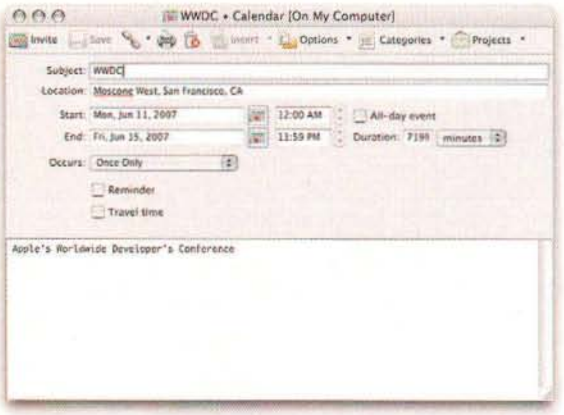
Figure 4. A Newly Created Contact

Please note that, in the example code above, AppleScript records were used to assign values to certain contact properties. This is because these properties reference classes that contain properties of their own. For example, the **business address** property of a contact references a value of class **postal address**. The **postal address** class possesses **street address**, **city**, **state**, **zip**, and **country** properties.

Working with Events

AppleScript may also be used to interact with calendar events in Entourage. Like outgoing messages and contacts, events may be created using the **make** command. For example, the following code will create a new event for this year's Worldwide Developer's Conference in San Francisco. Figure 5 shows an example of the newly created event.

```
set theStartDate to date "Monday, June 11, 2007 12:00:00 AM"
set theEndDate to date "Friday, June 15, 2007 11:59:00 PM"
tell application "Microsoft Entourage"
    make new event with properties {subject:"WWDC",
location:"Moscone West, San Francisco, CA", content:"Apple's
Worldwide Developer's Conference", start time:theStartDate,
end time:theEndDate}
end tell
--> event id 50 of application "Microsoft Entourage"
```



Subject: WWDC
Location: Moscone West, San Francisco, CA
Start: Mon, Jun 11, 2007 12:00 AM All-day event
End: Fri, Jun 15, 2007 11:59 PM Duration: 7199 minutes
Occurs: Once Only
Reminder
Travel time

Apple's Worldwide Developer's Conference

Figure 5. A Newly Created Event

In Conclusion

Like the other Microsoft Office applications, Entourage has fairly comprehensive AppleScript support, and we have really only scratched the surface in this month's column. You are encouraged to explore some of the other things that can be done via scripting, such as interacting with tasks, triggering AppleScripts from Entourage rules, and more. Be sure to view Entourage's AppleScript dictionary for a complete list of its AppleScript terminology.

Another great way to get started with scripting Entourage is to download some of the many freeware and shareware AppleScripts for Entourage that have been written by third-party developers, such as Paul Berkowitz. Many of these scripts are available from the ScriptBuilders section of MacScripter.net. http://scriptbuilders.net/

Note to Readers

Many of you who are regular MacTech readers know that I have been writing monthly AppleScript columns for MacTech for several years now. In addition to writing for MacTech, I've also been busy running Automated Workflows, LLC, my AppleScript and workflow automation consulting firm. As my company has grown, so have my commitments. For this reason, I have decided that it is time to begin focusing more on my company. Therefore, this month's column will serve as my final regular MacTech column.

Throughout the years, I have received numerous emails from you, the readers. I want to thank you for your many nice comments and questions along the way. As always, my email inbox will remain open, and you are welcome to continue sending your comments and questions to me in the future at ben@automatedworkflows.com. I will also be attending WWDC 2007, and I hope to have the opportunity to meet some of you there, as well.

For a complete list of my MacTech columns, please be sure to visit the "Archives" section of the MacTech website. You can also find links to all of my MacTech columns, as well as many of my other AppleScript and Automator articles, tips, techniques, and books on my website at http://www.automatedworkflows.com.

Keep scripting! MT

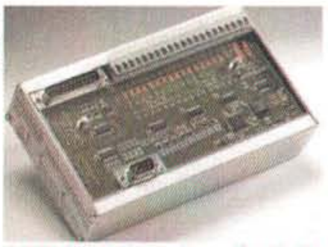
About The Author

***Ben Waldie is the author of the best selling books "AppleScripting the Finder" and the "Mac OS X Technology Guide to Automator", available from** http://www.spiderworks.com, **as well as an AppleScript Training CD, available from** http://www.vtc.com. **Ben is also president of Automated Workflows, LLC, a company specializing in AppleScript and workflow automation consulting. For years, Ben has developed professional AppleScript-based solutions for businesses including Adobe, Apple, NASA, PC World, and TV Guide. For more information about Ben, please visit** http://www.automatedworkflows.com, **or email Ben at** ben@automatedworkflows.com.*

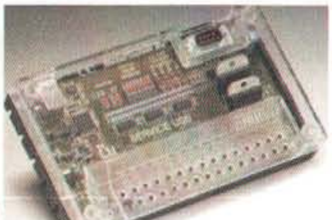
Control real-world devices

from your Mac with a wide variety of standard applications and programming languages such as:

- REALbasic
- AppleScript
- Runtime Revolution
- FileMaker
- Ragtime
- Xcode
- C/C++
- Cocoa
- Java
- CodeWarrior
- Carbon
- MaxMSP
- LabView
- DirectorMX
- 4th Dimension
- and the UNIX-Shell



SERVICE USB plus · $451.51



SERVICE USB · $296.33

Simply buy one interface from the SERVICE USB series and control motors, lamps, relays. Log values like temperature, pressure or brightness. Get the status of switches, light barriers or reed-contacts.

All interfaces ships with a complete SDK, drivers, sample apps and documentation.

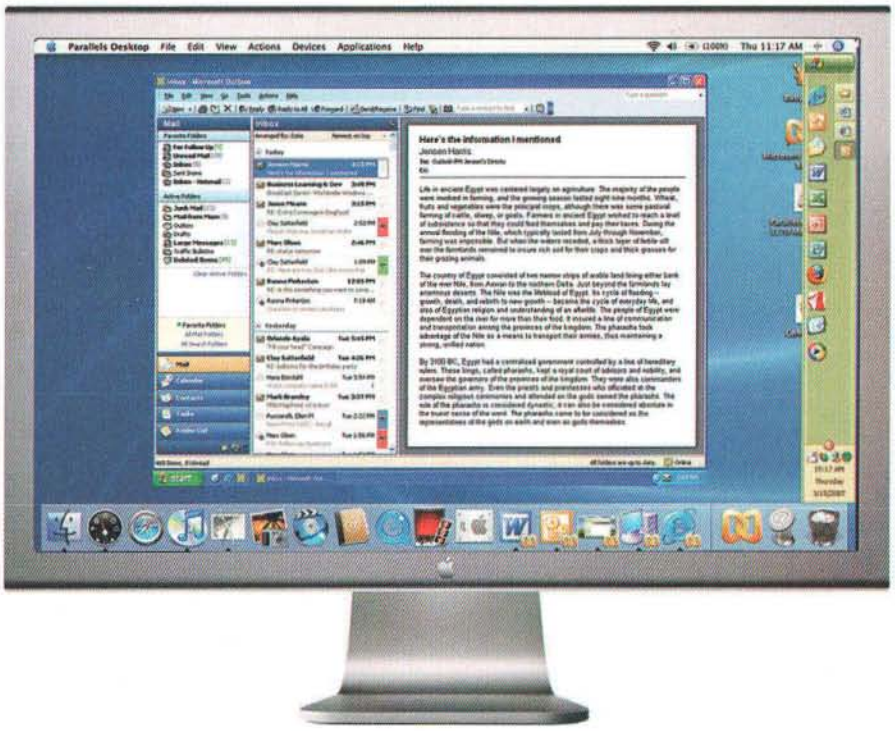
Custom made models, USB-POS accessories and special electronics are also available.

Well suited pre- and after sales eMail-support is available as a matter of course.

www.bkohg.com/products.html

Bönig and Kallenbach - GERMAN TECHNOLOGY

Instant compatibility.



Parallels™ Desktop for Mac running Microsoft® Outlook® using Coherence™

Run Windows® on your Mac® - without having to reboot.

"It's like having two computers in one, the best of both worlds".
- Walt Mossberg, Wall Street Journal

Parallels™ Desktop for Mac is the award-winning virtualization software that lets you run any combination of operating systems side-by-side with OS X on any Intel-based Mac system. Take advantage of Windows-only business-critical software, such as Outlook™ and SQL Server™, without giving up the security and reliability of your Mac. With the Coherence™ feature, you can even launch and run Windows applications directly from your OS X desktop.

Macworld BEST OF SHOW 2007

PC WORLD 20 MOST INNOVATIVE PRODUCTS

Download a trial today at parallels.com

parallels™

For more information about Parallels™ Desktop for Mac, please visit parallels.com or call us at 1-425 282-6504.

Website - www.parallels.com · Email - sales@parallels.com · Phone - 1-425 282-6504

Learn F-Script in 20 Minutes

And have fun playing with Core Image

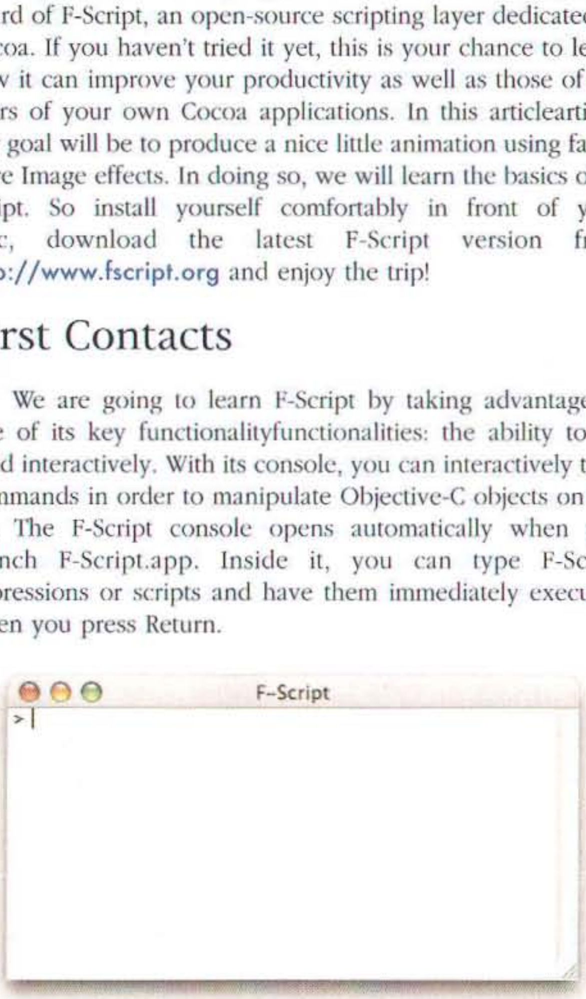
By Philippe Mougin

Welcome

If you are a Cocoa programmer chances are that you've heard of F-Script, an open-source scripting layer dedicated to Cocoa. If you haven't tried it yet, this is your chance to learn how it can improve your productivity as well as those of the users of your own Cocoa applications. In this articlearticle, our goal will be to produce a nice little animation using fancy Core Image effects. In doing so, we will learn the basics of F-Script. So install yourself comfortably in front of your Mac, download the latest F-Script version from http://www.fscript.org and enjoy the trip!

First Contacts

We are going to learn F-Script by taking advantage of one of its key functionalityfunctionalities: the ability to be used interactively. With its console, you can interactively type commands in order to manipulate Objective-C objects on the fly. The F-Script console opens automatically when you launch F-Script.app. Inside it, you can type F-Script expressions or scripts and have them immediately executed when you press Return.



```
> |
```

Figure 1. The F-Script console, waiting for your input

F-Script is a Smalltalk dialect for Cocoa and should look very familiar to you. Indeed, Brad Cox, who created objective-C describes it as "a hybrid language that contains all of C language plus major parts of Smalltalk". Here is an example of a message sending expression, both in Objective-C and F-Script. In this example, we ask for the current date using the NSDate class provided by Cocoa.

Objective-C	F-Script (i.e., Smalltalk)
[NSDate date]	NSDate date

As you can see, the expression is similar, except for the fact that, in F-Script, you don't have to put brackets around your message. This is because F-Script is a very simple language and sending a message is nearly the only thing you can do. Therefore, there is no need to have a special syntax to delimit messages. Now, if you type this expression in the console and hit Return, it will be immediately evaluated and the result will be displayed (obviously, the result you'll get will differ from the one shown below):

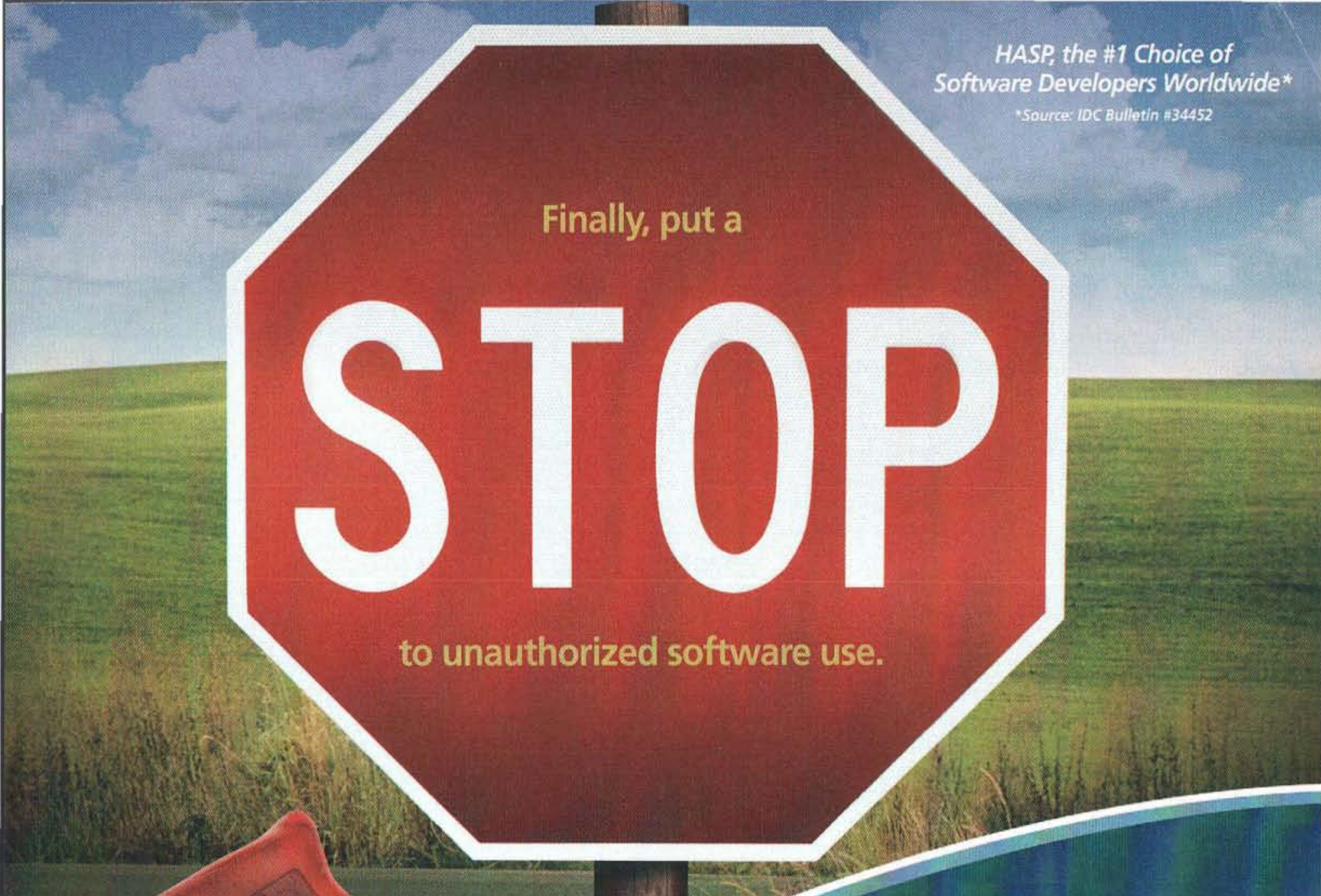
```
> NSDate date
2007-03-16 17:01:45 +0100
```

F-Script provides numerous tools to assist you during such interactive Cocoa sessions. In this first sessionsession, you are likely to find the following tips useful:

- The console keeps a history of your commands. You can navigate it using the up and down arrows of your keyboard. For instance, if you mistype something and F-Script signals an error, you can use this feature to get back at your command without having to retype it.
- You can insert a line break by pressing the Enter key (usually found on the numeric keypad) or by pressing Return while holding the Control key.
- The console also provides a code completion mechanism that you can use by pressing the F5 key. You can then navigate between arguments placeholders with Control-Slash.
- A graphical object browser opens automatically at startup. It is a very powerful tool with which you can explore objects and send them messages.

HASP, the #1 Choice of Software Developers Worldwide*

*Source: IDC Bulletin #34452



HASP HL

REINVENTING SOFTWARE PROTECTION & LICENSING

Follow the signs to HASP HL – the market's leading software DRM solution

HASP HL provides the highest level of security for your software. Featuring easy-to-use tools, HASP HL automatically protects your software and enables new and innovative licensing options.

- ***Strong copy protection:*** *Powerful 128-bit AES encryption provides a strong locking mechanism that ensures you get paid for every copy of your software in use.*
- ***Unmatched intellectual property protection:*** *Automatically protects Windows executables, DLLs, Mac applications, and offers class-level obfuscation for .NET framework 2.0 applications. Multiple encryption layers ensure your valuable IP remains unreachable and your R&D investment is protected.*
- ***Secure & flexible licensing:*** *HASP HL enables a variety of secure and flexible licensing models and ensures end-users comply with software licensing terms.*

See for yourself why IDC ranks HASP #1.

Request a Software Developer Kit today at www.Aladdin.com/StopPiracy_MT.

• NORTH AMERICA: 1-800-562-2543, 1-847-818-3800
• UK • GERMANY • ISRAEL • BENELUX • FRANCE • SPAIN • ASIA PACIFIC • JAPAN

Aladdin®

SECURING THE GLOBAL VILLAGE

www.Aladdin.com/HASP

Before continuing to talk about the language itself, let me give you a little bit of the history: Smalltalk was created in the early seventies at the famous Xerox Palo Alto Research Center, the PARC, by a team led by Alan Kay. As you might know, since then, Smalltalk has been having a big influence on the software industry. For instance, you might have heard about a visit that Steve Jobs made at the PARC in 1979. A visit that had a considerable influence on the design of the Lisa and the Macintosh computers. What Steve Jobs was shown there was Smalltalk. It had a graphical interface, was the first object-oriented system, and supported networking. "You guys are sitting on a gold mine here. Why aren't you making this a product?" asked the young Steve Jobs. A short time later, several people from the PARC were working at Apple and the rest is history...

So, what is the basic concept of Smalltalk? The key insight leading to the design of Smalltalk is that we can describe everything in terms of objects. As Alan Kay puts it "Smalltalk's design is due to the insight that everything we can describe can be represented by the recursive composition of a single kind of behavioral building block that hides its combination of state and process inside itself and can be dealt with only through the exchange of messages". Indeed, in Smalltalk, everything is an object, even numbersnumbers, or booleansBooleans.

It is also important to note that F-Script provides an interactive environment with which you can directly interact with your objects, instead of having to develop a specific application each time you want to do something.

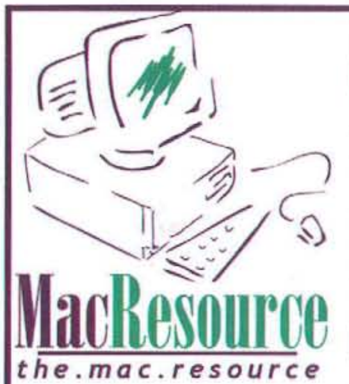
F-Script's Syntax

In a F-Script program the main control structure is message sending. In F-Script, as well as in Objective-C, a message with no argument is called a "unary message". A message with one or more colons in its selector is called a "keyword message". And, unlike Objective-C, there is a third kind of message in F-Script: a message that is composed of non-alphabetical characters like +, -, etc., is called a "binary message". A binary message always has only one argument.

Message type	Objective-C	F-Script
Unary	`[NSDate date]`	`NSDate date`
Keyword	`[NSDate dateWithTimeIntervalSinceNow:10]`	`NSDate dateWithTimeIntervalSinceNow:10`
Binary	Not available	`date1 < date2`

The Objective-C equivalent to `date1 < date2` would be `[date1 compare:date2] == NSOrderedDescending`.

As in Objective-C, messages can be chained together. Expressions are evaluated from left to right, giving us the same semantics, as shown below.



MacResource
the.mac.resource

Wireless Products

Airport Cards
Extreme: $69.99
Standard: $199.99
Routers: $59.99

iPods Galore!

iPod Minis from $129.99
20 Gig from $159.99
30 Gig from $179.99
40 GIG from $199.99
60 GIG from $239.99

Parts for Every Mac System!

Logic Boards

G3/G4 PCI/AGP : $129
G4 Gigabit/D Audio: $199
G4 Quicksilver: $349
G4 MDD: $489
G3 iMac Logics From $69
G4 iMac Logics From: $199
G5 Towers.....Call
G4 Xserves $149-$299
G5 xserves $599

Power Supplies

G4 iMac 15/17/20: $79/$99/$119
G5 iMac 17/20: $149/$179
G5 Tower: $169/$199
G4 Qsilver/D audio:$179
G4 Gig E-net/MDD: $199
G4 AGP 208/237w: $179/$219
Logic board and power supplies require exchange.

G4/G5Processors

We carry Original Processors for most G4s, G5s and Xserves
Call for best pricing

Thousands of parts in stock for all Mac systems!

1-888-Mac-Resource

www.mac-resource.com

Mac Systems

G4 733Mhz	$599
G4 800Mhz:	$649
G4 867/933Mhz:	$899
G5 1.6Ghz:	$899
G5 1.8Ghz:	$1099
G5 1.8Ghz DP	$1249
G5 2.0Ghz DP	$1399

New Systems Arriving Daily! Call for latest Stock.

NEED G4, G5 OR INTEL IMACS?
iMac G4s $399,iMac G5 1.9Ghz 17" $999
1.8Ghz 17"/20": $859/$999
Intel Dual Core Demos from $849!!!

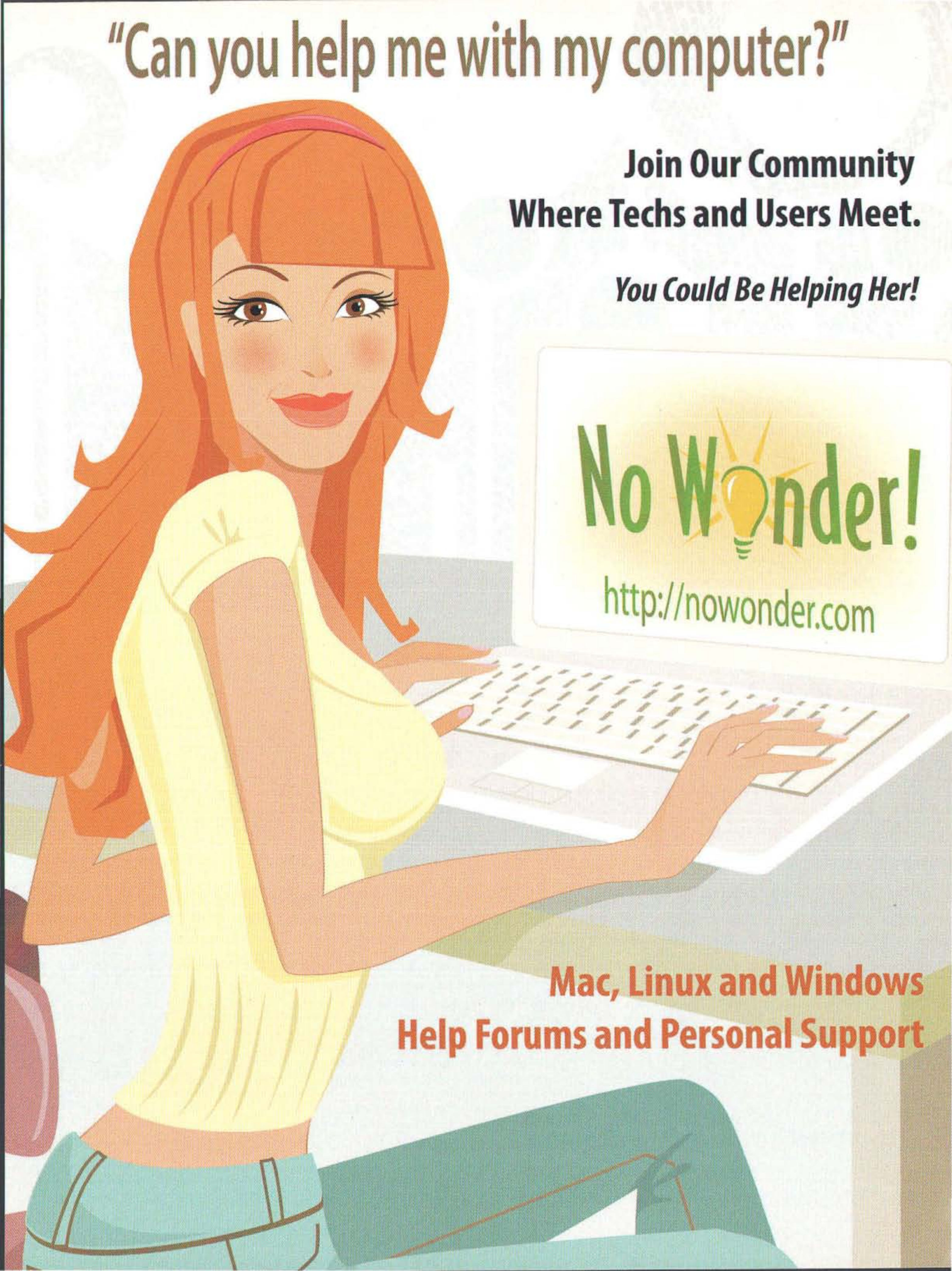
WE HAVE G5 XSERVES AND RAIDS EVEN IF APPLE DOESN'T!!!!

G4 XSERVES STARTING@ $SOLD OUT!!!!
G5 XSERVES STARTING @$1499!!!!
1 TB XSERVE RAIDS FROM $3499!!!!
2.8 TB XSERVE RAIDS FROM $4499!!!!
5.6 TB XSERVE RAIDS FROM $6999!!!!
3.5 TB/7.0 TB XSERVE RAIDS $6499/9599
RAID CARDS, FIBRE CARDS, DRIVE MODULES POWER SUPPLIES, CONTROLLER MODULES ETC, OVERNIGHT SERVICE AVAILABLE!!!!

Displays

REFURBISHED DISPLAYS
Apple 17" Studio LCD: $299
Apple 23" Cinema: $899
Apple 22" Cinema: $799
Apple 15-inch Studio LCD: $149
Apple 17" Studio CRT,ADC or VGA: $69.99

Products may be refurbished or demo, call for more information


"Can you help me with my computer?"
Join Our Community
Where Techs and Users Meet.
You Could Be Helping Her!
No Wonder!
http://nowonder.com
Mac, Linux and Windows
Help Forums and Personal Support

Objective-C	F-Script
`[[NSDate date] timeIntervalSinceNow]`	`NSDate date timeIntervalSinceNow`

But sometimes, we need a way to determine the order of evaluation of messages. F-Script introduces a precedence rule (the only precedence rule in the language): unary messages are evaluated first, then binary messages, and then keyword messages. If you want to change the order of evaluation, you can use parenthesis to delimit a message.

The following example shows a few other differences between F-Script and Objective-C:

Objective-C	F-Script
`NSDate *date1 = [NSDate date];`	`date1 := NSDate date.`

As you see, there is no type declaration in F-Script. Everything is an object and variable need not be explicitly typed. The assignment syntax uses **:=** instead of just = in Objective-C, and the instruction separator is not the semicolon, but the period symbol, like in English sentences. The following table shows other differences. As you can see, strings are enclosed in single quotes, and comments are enclosed in double quotes.

Objective-C	F-Script
`@"A string"`	`'a string'`
`/* A comment */`	`"A comment"`
`@selector(dateWithTimeIntervalSinceNow:)`	`#dateWithTimeIntervalSinceNow:`
`[NSMutableArray arrayWithObjects:@"Hi", @"mom", nil]`	`{'Hi', 'mom'}`
`NSMakePoint(200, 80)`	`200<>80`

Displaying a picture on screen

We now know enough of F-Script to begin with our Core Image program. We will first create an NSURL object referring to the image we want to display. In this exampleexample, we will use an image that is stored on disk in the desktop picture folder. You can type the code below in the F-Script console to have it executed immediately.

```
imageLocation := NSURL fileURLWithPath:'/Library/Desktop
Pictures/Nature/Clown Fish.jpg'.
```

The `imageLocation` variable now points to our NSURL object. We will now create a CIImage object, initialized with our image on-disk.

```
image := CIImage imageWithContentsOfURL:imageLocation.
```

WASTE 3.

The powerful text engine for your Carbon applications.

WASTE is a fully Unicode–savvy text engine for Mac OS X, suitable for a large variety of applications requiring advanced **text display and editing** functionality.

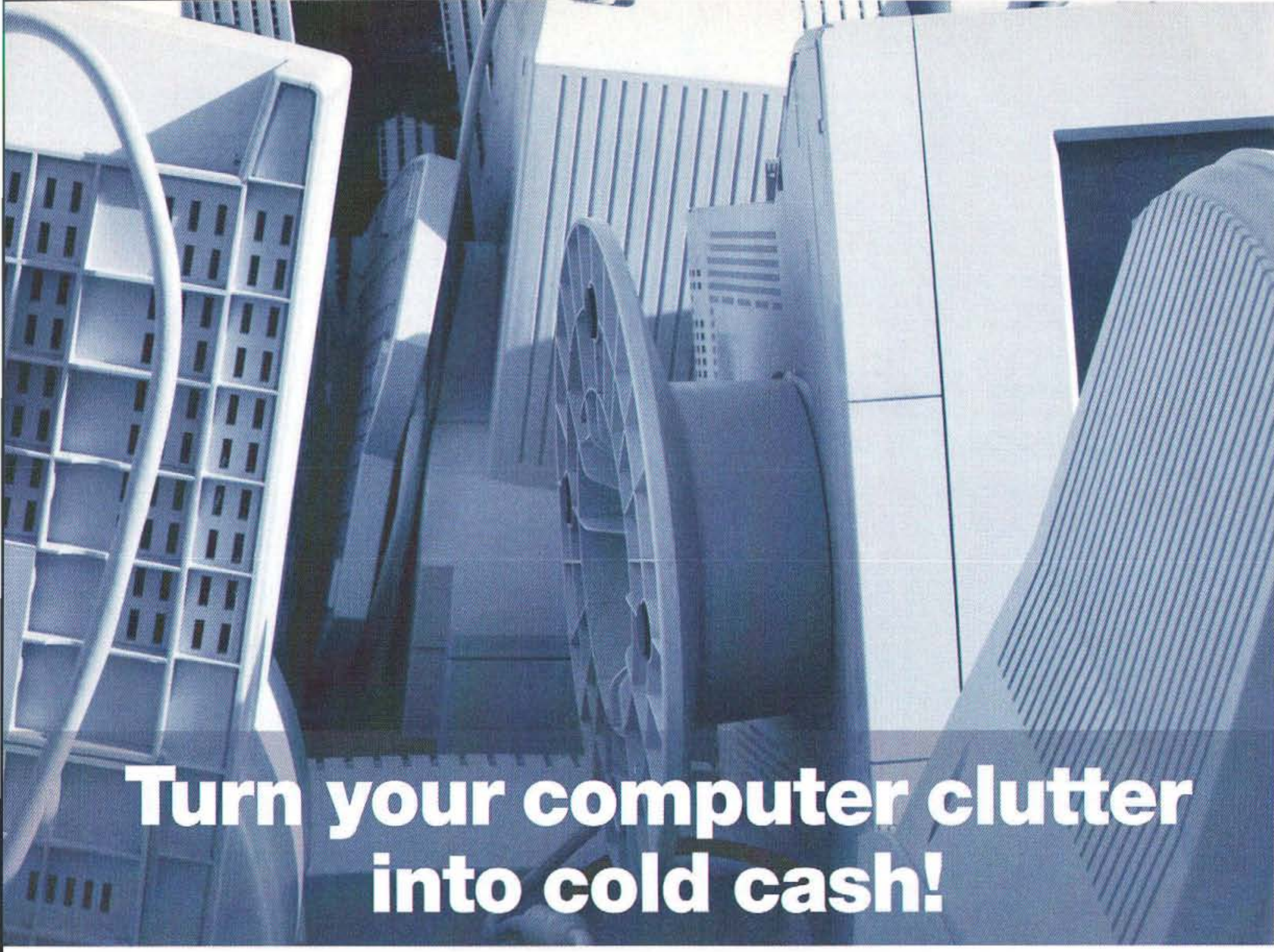
Extremely fast, WASTE is a Universal framework built on modern technologies such as **Unicode**, **ATSUI** and **Quartz 2D**. Plus, it features unlimited undo/redo, **RTF** support, **dictionary** lookup, **PDF** generation, and much more.

Many developers have replaced their TextEdit– or HITextView–based code with WASTE in a few hours.

Whether you need to display a few lines of text in a graphics application, or add a full–blown text editor to your product, WASTE is for you.

Source code license available. Check it out now at **www.ovolab.com/waste**.

OVOLAB
software for the creative mind



WE BUY Apple Macintosh and other brand name laptops, desktops and servers. We buy from businesses, institutions and individuals. One unit or a thousand, working or not, contact us for a quote. **AND...**

WE SELL fully refurbished Apple Macintosh computers and other brand name laptops, desktops and servers. We provide you a new computer experience, at a used computer price.

Call us today

(800) 960-1140

MicroReplay

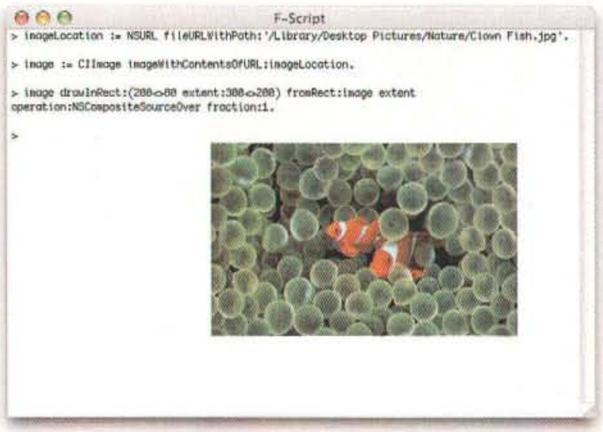
www.microreplay.com

MicroReplay, Inc. 309 Walnut Street, Wellesley, MA 02481 - (781) 237-1140

Note that we are using standard methods provided by the Mac OS X frameworks. Now that we have an image object, we can ask it to draw itself on screen, again using a standard method.

```
image drawInRect:(200<>80 extent:300<>200) fromRect:image
extent operation:NSCompositeSourceOver fraction:1.
```

After executing this code, we should see a beautiful little image displayed in the console, as shown below.



```
F-Script
> imageLocation := NSURL fileURLWithPath:'/Library/Desktop Pictures/Nature/Clown Fish.jpg'.

> image := CIImage imageWithContentsOfURL:imageLocation.

> image drawInRect:(200<>80 extent:300<>200) fromRect:image extent
operation:NSCompositeSourceOver fraction:1.

>
```

Figure 2. Loading and displaying an image using Core Image and F-Script

The first argument passed to the drawing method is the rectangle we want to draw in, which is denoted with `200<>80 extent:300<>200`. This expression actually creates an NSValue object representing a rectangle with an origin at (200, 80), a width of 300 and a height of 200. When passed to the method, the NSValue is automatically mapped by F-Script to an NSRect structure. This kind of automatic mapping between objects and primitives Objective-C types makes it possible to use the Mac OS X Objective-C based frameworks from a pure object language such as F-Script. You can change the rectangle size to make the image bigger or smaller and immediately see the result on-screen.

The drawing method draws the image in the current graphic context, which, in our example, happens to be the F-Script console. It is possible, of course, to draw elsewhere, using standard Mac OS X techniques.

Using core image filters

Core Image filters allow us to do all kind of highly optimized image processing. Mac OS X comes bundled with dozens of filters. Going forward with our exploration, we will apply a filter known as CIBumpDistortion, which creates a bump in the image. You are encouraged to try other filters as well. F-Script's interactivity makes it fun and efficient to explore such Mac OS X capabilities. The following F-Script code creates a CIBumpDistortion filter object and configures it to process our image, creating a bump of radius to 800 and of scale 2.

Attention: Corporate and Independent Software Vendors

Alternative to FileMaker?

Yes! It's SERVOY®

Are you struggling with your FileMaker application - do you need it to scale to a larger audience, connect to SQL databases, run faster and with less downtime? Would you like to make changes and not have to import data at night or on the weekends? You need to check out Servoy. While FileMaker is a great tool for some small, simple jobs, Servoy is an excellent tool when you need to grow to the next level.

We've been around since 1998 and have helped customers such as UCLA, Motorola, Proctor & Gamble, Turner Broadcasting, US Geological Survey, Wells Fargo, City of Hope, Sony, Stanford Medical and hundreds of other firms. Over 10,000 developers now program with Servoy in over 25 countries around the globe.

Some of our happiest customers are Independent Software Vendors (ISVs) who wrote a great software package in FileMaker for a vertical market - and now need to upgrade it for reduced maintenance, painless upgrades, and greater functionality for Web 2.0 (e.g. - offering your product as a "rented" application or Software as a Service, adding AJAX capabilities, and integrating with Google applications).

You owe it to yourself to evaluate your alternatives. Have a look at Servoy today, and check out the resources available just for FileMaker users.

So if you have 100 users or more, visit us at **www.servoy.com/fmp** to see what Servoy can do for you as a FileMaker user.

STOP SHARING!

START FAXING!

Each subscriber receives faxes directly by email as PDF file attachments.

Corporate accounts from 3 to 100+ users available

For more information and a special offer for MacTech readers, visit

www.MaxEmail.com/MacTech

maxemail®

Call: 800-964-2793

```
filter := CIFilter filterWithName:'CIBumpDistortion'.
filter setValue:image forKey:'inputImage'.
filter setValue:(CIVector vectorWithX:1000 Y:700)
forKey:'inputCenter'.
filter setValue:900 forKey:'inputRadius'.
filter setValue:1 forKey:'inputScale'.
```

Now that the filter is configured, it will apply itself to our image when asked to provides its output, creating a new image and giving it back to us:

```
bumpedImage := filter valueForKey:'outputImage'.
```

We can now draw this new image on screen:

```
bumpedImage drawInRect:(200<>80 extent:300<>200)
fromRect:image extent operation:NSCompositeSourceOver
fraction:1.
```

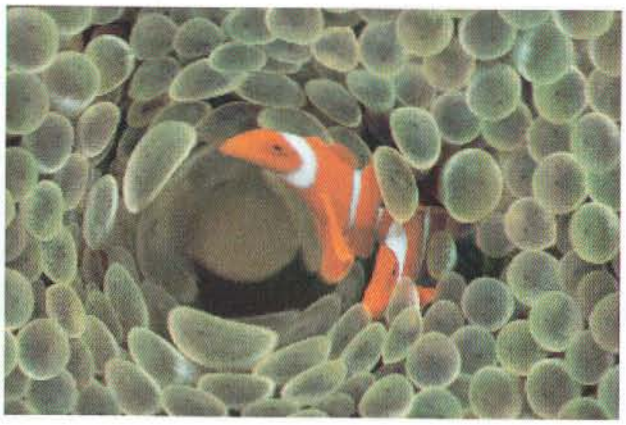


Figure 3. Our image after processing by a Core Image "bump" filter

To understand how the filter works, it is interesting to change its configuration (for instance, the values of its radius and its scale) and to regenerate and redisplay the image. If you are sitting behind an F-Script console, you are encouraged to do so!

Using blocks to create an animation

Now that we know how to process and display an image, we can create a nice little animation by repeatedly processing the image with a varying filter and displaying the result. To do that we just need to learn how write a loop using F-Script.

But, wait a minute... Isn't F-Script supposed to have a very simple syntax, where everything is expressed by sending messages to objects? Well, this is exact and, in fact, F-Script does not have any special syntax for control structures such as loops or conditionals. So the question here is "How can we express useful programs without such syntax?" To answer that, let me introduce you to the concept of code blocks in F-Script. Below, we see a code block in Objective-C and one in F-Script. Note the use of brackets in F-Script, instead of curly braces.

CodeMeter A Safe Shell

No.1 in Software and Document Protection!

Highest Security

2007 SIIA //CODiE// FINALIST

- Vendor selectable secret and private key.
- Strong encryption algorithms with AES 128-bit and ECC 224-bit.
- Best-in-class tools for automatic protection (envelope, without source modification) for Win32, Win64, .NET, Java and MacOS X Universal (PPC, Intel).

Best Flexibility

- More than 1000 independent licenses can be protected by one CM-Stick.
- One versatile hardware key for all license models including floating network licenses.
- Multi platform support including Windows, MacOS X and Linux.

New Distribution Channels

- License transfer by SOAP based CM-Talk or file based Field Activation Service in e-shops.
- Multiple-purpose, including protecting low cost software and digital content.

Unique End User Advantages

- First and smallest dongle with up to 2 Gbyte flash drive.
- No drivers necessary – can be used without administrator rights.
- CM Password Manager, secure virtual drive and secure login.

Order your Free Software Development Kit now!
Phone 1-800-6-GO-WIBU | order@wibu.us

2005 SIIA //CODiE// FINALIST
2005: Best Digital Rights Management Solution: Software

2006 SIIA //CODiE// FINALIST
2006: Best Security Software

The very highest quality by WIBU-SYSTEMS!
More than 3000 application developers and creators of intellectual property trust in WIBU-SYSTEMS solutions since 1989.

WIBU SYSTEMS

WIBU-SYSTEMS USA Inc.
2429 NW 197th Street
Shoreline, WA 98177, USA
www.wibu.com
info@wibu.us
Tel: 1.800.6.GO.WIBU
1.206.546.4891
Fax: 1.206.237.2644

seefile™

LIVE FILE SHARING+
ONLINE APPROVALS

WHAT IS THIS THING CALLED SEEFILE?

Affordable, Apache-based webserver software, accessible from all major Mac and PC browsers, for customer image reviews and approvals.

SeeFile lets you maintain private logins to different folders, and manages all permissions for uploads, downloads, previews, watermarks and messaging.

- pro photography
- prepress
- archives
- videos

www.seefile.com
30 day working demo available
call 1-617-262-2464 for reseller info

Objective-C	F-Script

`{`
`  instruction1;`
`  instruction2;`
`}`
	`[`
	`  instruction1.`
	`  instruction2.`
	`]`

The code blocks look similar, but the way they work is quite different. In Objective-C, when the computer executes the code block, it simply executes the instructions in it immediately. In F-Script, the code block is actually a kind of literal notation for an object that contains the instructions. In other words, a block represents a deferred sequence of actions. In F-Script, the presence of a code block does not lead to the execution of its content, but to the creation of a block object, that can then be asked to execute the instructions. To do that, we send the "value" message to the block. The result returned by the execution of a block is the result of the evaluation of its last instruction.

As you can see below, F-Script blocks can have local variables, just like in Objective-C. If the instructions in the block refer to a variable that is not declared as local, F-Script will look for it in the enclosing lexical context of the block, as is the case in Objective-C.

Objective-C	F-Script

`{`
`  id local1, local2;`
 	
`  instruction1;`
`  instruction2;`
`}`
	`[`
	`  |local1 local2|`
	
	`  instruction1.`
	`  instruction2.`
	`]`

The main point to understand here is that F-Script blocks are objects. Like with any object, you can send messages to a block, you can assign a block to a variable, store a block in a collection, pass a block as an argument to a method, archive a block on-disk, and so on. Blocks are not unique to F-Script (or Smalltalk). They are present in numerous languages (sometimes under the name of "closure" or "lambda expressions") such as Ruby, Python, Lisp, GroovyGroovy, and the forthcoming C# 3.

Now that we have blocks, it is easy to do conditional evaluation. Boolean objects provide a method named **ifTrue:** which takes a block as argument. If the value of the Boolean is true, then the block is executed by the method.

Objective-C	F-Script

`if (a > b)`
`{`
`  `*instructions*
`}`
	`(a > b) ifTrue:`
	`[`
	`  `*instructions*
	`]`

YOUR ULTIMATE 3D AQUARIUM!
NEW RELEASE!
FORTY BEAUTIFUL FISHES,
TURTLES, JELLIES, & SHARKS!
INTERACTIVE SCREENSAVER
TAP THE GLASS, FEED THE FISH
AQUAZONE SEVEN SEAS DELUXE
Includes Sharks!
40 fish inside! Have as many onscreen as you like!
Allume Systems
A Division of Smith Micro Software

A virtual aquarium so vibrant, you'll swear the fish are real! Available from your favorite retailers and catalogs.

Aquazone Seven Seas Deluxe includes 40 fresh and saltwater fish, jellyfish, adult and baby sea turtles, and sharks! Have entire schools of fish on screen swimming through 20 customizable 3D tanks!

Boolean objects also have a method named **ifTrue:ifFalse:** that lets you have something equivalent to the if/else control structure of Objective-C. This method takes two blocks as arguments. One that gets executed if the booleanBoolean is true and the other one that gets executed if the booleanBoolean is false.

Objective-C	F-Script

`if (a > b)`
`{`
`  instructions`
`}`
`else`
`{`
`  instructions`
`}`
	`(a > b) ifTrue:`
	`[`
	`  instructions`
	`]`
	`ifFalse:`
	`[`
	`  instructions`
	`]`

For performing our animation, we need a way to repeatedly evaluate a block. Let's review how F-Script provides this.
Blocks provide a method named **whileTrue:**, which takes another block as argument. The receiver of the **whileTrue:** message evaluates itself, and, if the result of this evaluation is a booleanBoolean with a value of true, the argument gets evaluated. This process is repeated as long as the receiver evaluates to true.+

Note that in the example with conditionals, the **ifTrue:** message was sent to a booleanBoolean object. In the latest example, the **whileTrue:** message is sent to a block that returns a booleanBoolean. This is very different. Indeed, it would not make sense to implement a **whileTrue:** method in the booleanBoolean class. This is because the value of a particular booleanBoolean never changes; whereas the value returned by the evaluation of a block can change from one evaluation to another.

Now that we know how to express repetitive evaluation, we can finally write our animation:

```
keyWindow := NSApplication sharedApplication keyWindow.
rect := (200<>100 extent:300<>200).
i := 0.

[i < 2500] whileTrue:
[
    filter setValue:(CIVector vectorWithX:i Y:700)
forKey:'inputCenter'.
    bumpedImage := filter valueForKey:'outputImage'.
    bumpedImage drawInRect:rect fromRect:image extent
operation:NSCompositeSourceOver fraction:1.
    keyWindow flushWindow.
    i := i + 5.
]
```

As you can see, we move the bump across the image, by varying the X component of the CIVector object that define the center of the bump. We use a control variable named "i" that we increment by five at each iteration of our loop until it becomes equal to 2500. We also ask the window to flush itself at each step of our iteration in order to display the new image and produce the animation effect.

Playing catch up?
Not anymore.

KerioMailServer6

Your email, address books, calendars and tasks. All synchronized with Entourage and Outlook and shared with your colleagues. With integrated anti-spam and McAfee virus filtering.

Finally free to work as a team.

www.kerio.com

KERIO

BookEndz®

DOCKING STATIONS FOR APPLE COMPUTERS

Convert your MacBook Pro®, MacBook®, PowerBook®, or iBook® into a desktop in seconds without misplacing cables or damaging connectors.

15.4" MacBook® Pro Docking Station

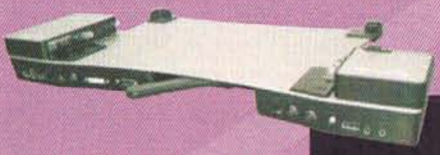
- Aluminum Plate helps in cooling of notebook
- Connectors are routed to rear of Dock
- MagSafe® Housing prevents accidental disconnect

13.3" MacBook® Docking Station

- Also available in Black
- 5 USB 2.0 compliant ports (4 port powered or unpowered hub, DC adapter included)
- Built in DVI/VGA full size
- Gigabit Ethernet RJ45

17" PowerBook® G4 Docking Station

12" PowerBook® Docking Station



15" PowerBook® G4 Docking Station

iBook® G3/G4 Models Docking Station

Visit our website for latest product announcement www.BookEndzdocks.com

BookEndz®

Manufactured by Olympic Controls

1250 Crispin Drive • Elgin, Illinois 60123 • USA

Phone: 847-742-3566 • Fax: 847-742-5686 • Toll Free: 888-622-1199

E-mail: Sales@BookEndzdocks.com

Blocks with arguments

For such kind of iteration, however, a *for loop* is more appropriate, and you might wonder if F-Script provides it. Well, it does! But in order to master it we must learn another feature of blocks: support for arguments. Block arguments are declared at the beginning of the block, just after the opening bracket. Each argument name is specified after a colon and a vertical bar ends the argument list. A block must then be executed using an appropriate "value..." message. For example, here is a block with no argument. We evaluate it by sending it the `value` message.

`['hello world'] value`	returns	`'hello world'`

Below is a block with one argument. In this case, we send a `value:` message, specifying the argument that will be passed to the block.

`[:a| a class] value:'a string'`	returns	`String`

Here is a block with two arguments. To evaluate it, we send it a `value:value:` message.

`[:a :b| a + b] value:2 value:3`	returns	`5`

R A D I O T O P E
TECHNOLOGY CONSULTING

Aligning business goals and technology.
Bringing enterprise solutions to small and medium businesses.

email: info@radiotope.com
http://www.radiotope.com

Now that we have blocks with arguments, we can make use of powerful methods. For example, numbers have a method named `to:do:` which takes an number and a block as arguments. The block is evaluated for each integer between the receiver and the first argument (both included).

Objective-C	F-Script

`for (int i=0; i <= 100; i++)`
`{`
`  instructions using i`
`}`
	`0 to:100 do:`
	`[:i|`
	`  instructions using i`
	`]`

F-Script also provides a `to:by:do:` method that let us specify an iteration step.

Objective-C	F-Script

`for (int i=0; i <= 100; i = i + 5)`
`{`
`  instructions using i`
`}`
	`0 to:100 by:5 do:`
	`[:i|`
	`  instructions using i`
	`]`

We can make use of it in our animation script, which then becomes:

```
keyWindow := NSApplication sharedApplication keyWindow.
rect := (200<>100 extent:300<>200).

0 to:2500 by:5 do:
[:i|
    filter setValue:(CIVector vectorWithX:i Y:700)
forKey:'inputCenter'.
    bumpedImage := filter valueForKey:'outputImage'.
    bumpedImage drawInRect:rect fromRect:image extent
operation:NSCompositeSourceOver fraction:1.
    keyWindow flushWindow.
]
```

You can change the value of the step to see how it makes the animation faster or slower.

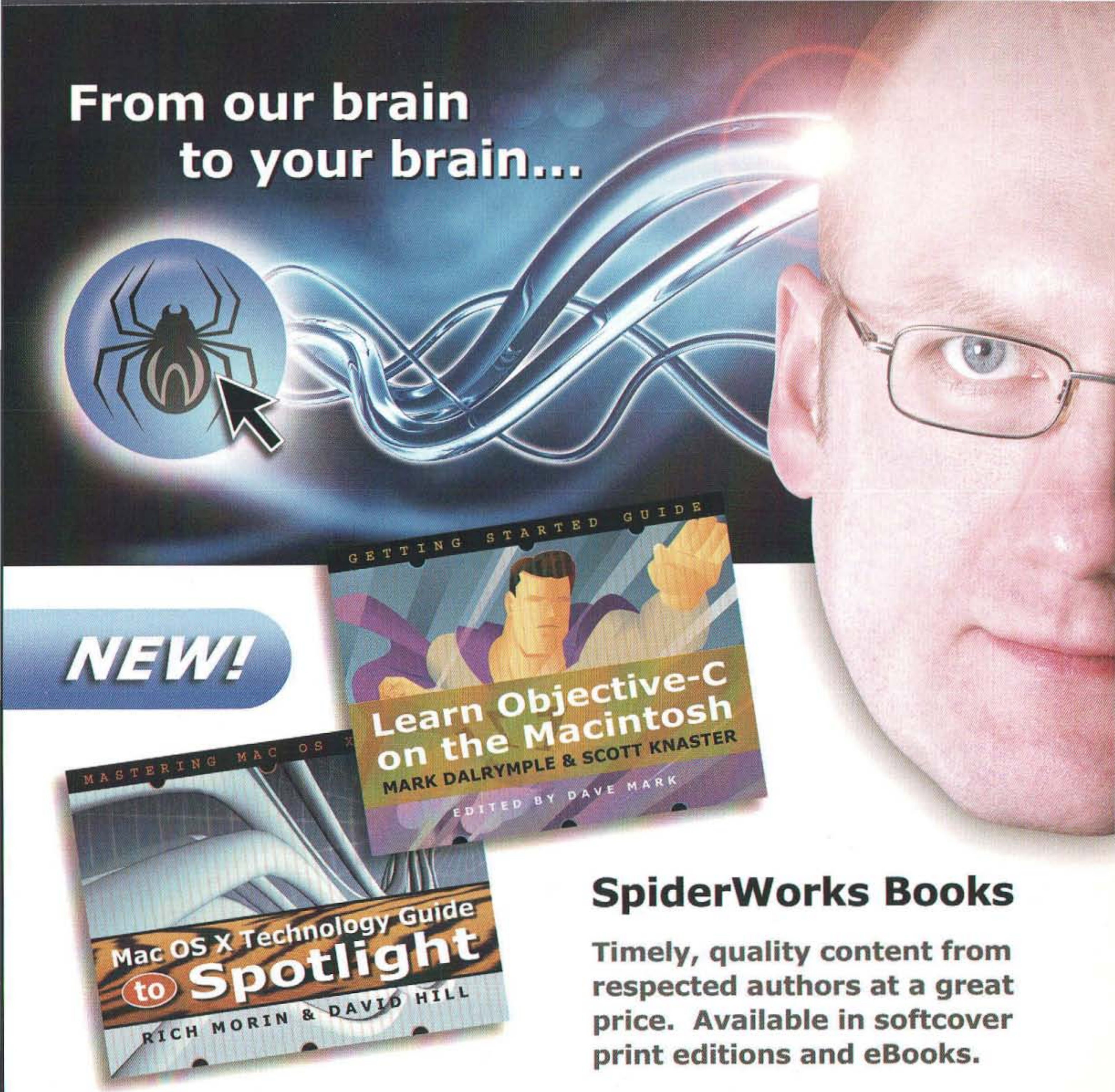
Conclusion

Now that you are familiar with F-Script, you can use it whenever you want to explore a new Objective-C API, interactively prototype code or debug an application. And since you can easily embed it into your own application, you can provide your users with an interactive and scripting layer for your application's functionalities, by just exposing them as Objective-C objects. MT

About The Author

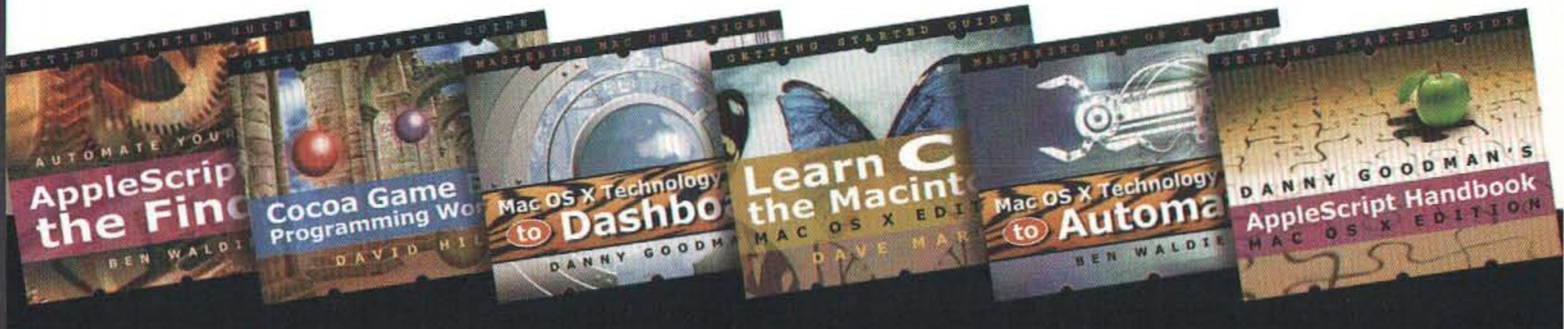
Philippe Mougin is the creator of F-Script. He works at OCTO Technology, a French consulting company, where he explores and promotes the use of dynamic languages in enterprise systems. You can reach him at pmougin@acm.org.

From our brain to your brain...


NEW!
GETTING STARTED GUIDE
Learn Objective-C on the Macintosh
MARK DALRYMPLE & SCOTT KNASTER
EDITED BY DAVE MARK
MASTERING MAC OS X
Mac OS X Technology Guide to Spotlight
RICH MORIN & DAVID HILL

SpiderWorks Books

Timely, quality content from respected authors at a great price. Available in softcover print editions and eBooks.



Download FREE Previews at SpiderWorks.com

By Edward Marczak

The man Show

Learning shell utilities with and without man

Introduction

Documentation. The ugly reality is that it's usually an afterthought for a project, if completed at all. We like to code, we like to connect systems, but rarely do we like to document the work. *man* ("manual") *pages*, the built-in documentation system, have thousands of entries for shell utilities. Depending on the author, these entries range from well-written, humorous and pleasurable reading, down to sparse, terse and frustrating. Sometimes, the simple act of giving usage examples would make all the difference in usefulness and clarity. While I can't cover every shell utility here, I would like to point out some that you should know about, but that may not have the best documentation or just lack examples.

networksetup

Just as the name implies, **networksetup** is a utility to configure the network interfaces on a Mac OS X machine. There's no man page for this one at all. By default, it's not even in your path. Currently, under OS 10.4 ("Tiger"), you'll find **networksetup** located at: /System/Library/CoreServices/RemoteManagement/ARDAgent.app/Contents/Support/networksetup (whew!). There is, though, a usage statement printed if you run the command but lists no options and provides no examples.

Since OS X uses **configd** to inform the system of its current configuration state, the old-school utilities, such as **ifconfig** don't really work too well. Oh, yes, they can *read* the current state of the network, but setting it is another matter. This will work for a little bit, at least until configd receives a change event and reconfigures things for you.

networksetup uses terminology that is closer to the OS X GUI than traditional devices, too. First thing you typically need to figure out are the names of the network services being offered – a.k.a. the interface names:

```
# networksetup -listallnetworkservices
An asterisk (*) denotes that a network service is disabled.
Bluetooth
Built-in Ethernet
Built-in FireWire
AirPort
Parallels Host-Guest
Parallels NAT
VPN (L2TP)
```

(You do, by the way, need admin level privileges to run this utility). You'll note that these names correspond to the names in the Network System Pane:

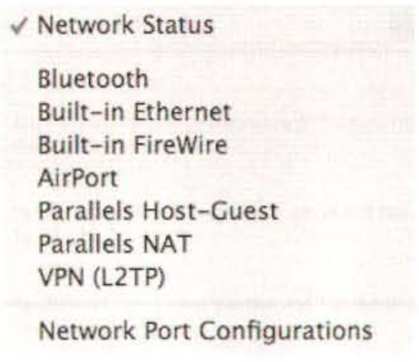


Figure 1 – Interface names

If you rename the interface in the Network Pane, networksetup will see it that way, too. A possibly more useful list, if you're on a machine that you're not familiar with yet, is gained from the –listnetworkservice order switch. It shows both the interface name *and* the order that OS X uses it:

```
# networksetup -listnetworkserviceorder
An asterisk (*) denotes that a network service is disabled.
(1) Bluetooth
(Hardware Port: Bluetooth, Device: Bluetooth-Modem)

(2) Built-in Ethernet
(Hardware Port: Built-in Ethernet, Device: en0)

(3) Built-in FireWire
(Hardware Port: Built-in FireWire, Device: fw0)

(4) AirPort
(Hardware Port: AirPort, Device: en1)

(5) Parallels Host-Guest
(Hardware Port: Parallels Host-Guest, Device: en2)

(6) Parallels NAT
(Hardware Port: Parallels NAT, Device: en3)

(7) VPN (L2TP)
(Hardware Port: Windows L2TP, Device: )
```

This list is a great reminder that routes may be overridden by the order of interfaces.

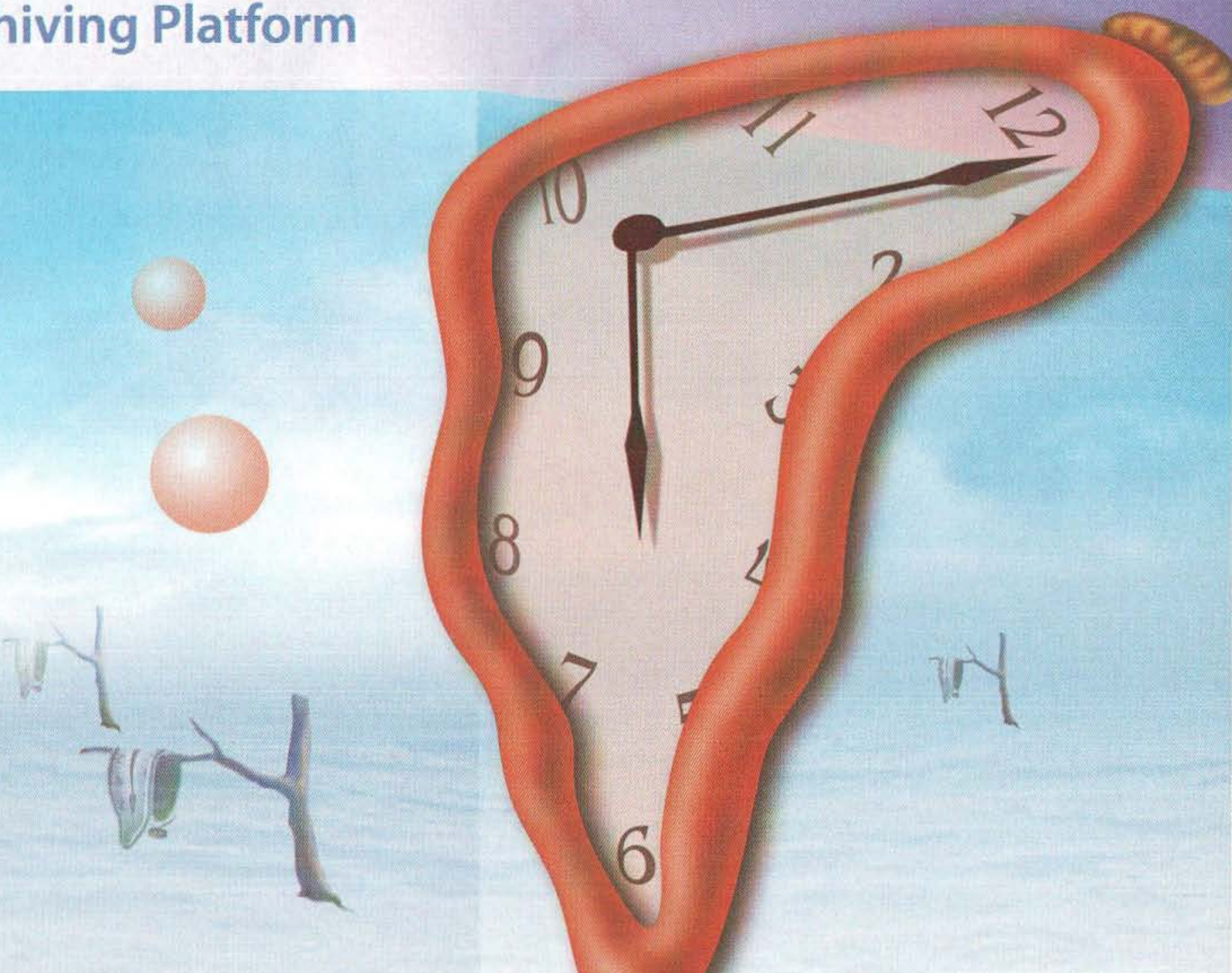
Well, how about setting up the device? There are switches that act like each of the settings in the GUI:

Atempo

Protect. Archive. Restore.

Ever Wish You Could Turn Back the Clock?

Mac Backup
Heterogenous Server Protection
Archiving Platform



For a Free Trial of Time Navigator Please Visit
www.atempo.com/mactech** or call **866.417.0200

Atempo

Protect. Archive. Restore.

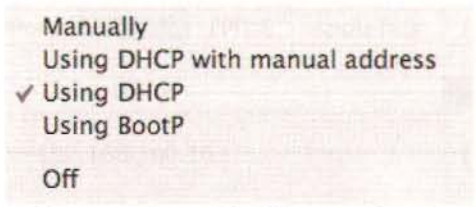

Manually
Using DHCP with manual address
✓ Using DHCP
Using BootP
Off

Figure 2 – How d'ya like your interface?

Probably of most interest is "-setmanual". To configure the built-in Ethernet interface to have an IPv4 address of 192.168.50.77, with a subnet mask of 255.255.255.0 and a default gateway of 192.168.50.1, you'd use this:

```
networksetup -setmanual "Built-in Ethernet" 192.168.50.77
255.255.255.0 192.168.50.1
```

One nice trick that doesn't seem apparent from the usage help, but has worked for me (up through 10.4.9), is that the router is *optional*. This is critically important when you're configuring an interface that will be active along with another.

The second option in our list is "-setmanualwithdhcprouter". This is just like "-setmanual", without being able to specify the subnet or router:

```
networksetup -setmanual "Built-in Ethernet" 192.168.50.77
```

Setting an interface to use DHCP is simple:

```
networksetup -setdhcp "AirPort"
```

You can also supply a client id after the interface name, if necessary. You can also clear the current client id by using "Empty" as the client ID name.

Similar to "-setdhcp" is "-setbootp". The command is the same as "-setdhcp", however, bootp doesn't support a client id, so, nor does this command.

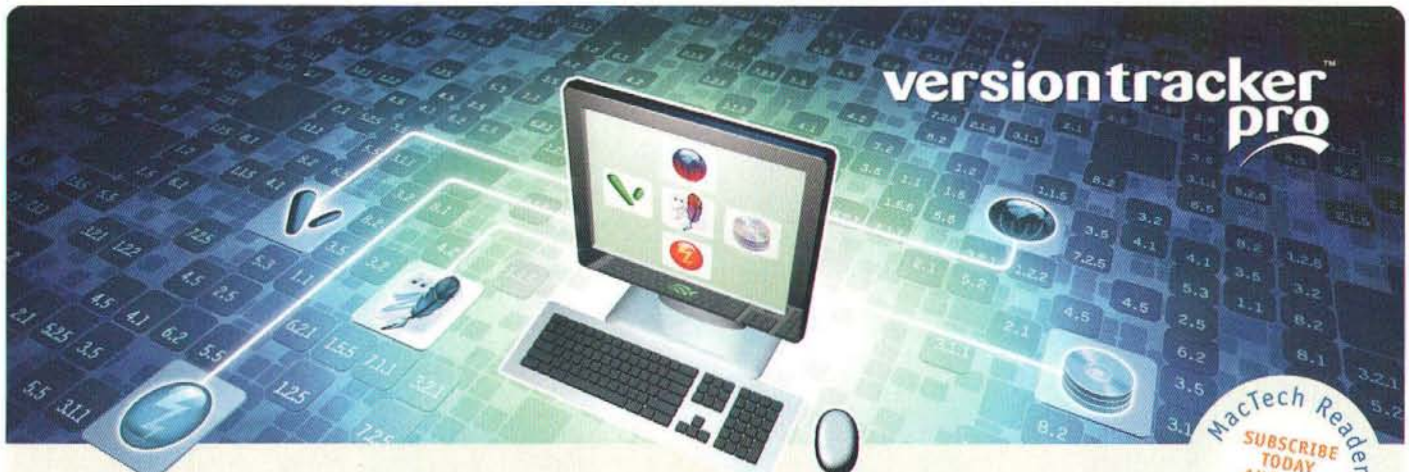
Finally, "-setnetworkserviceenabled" roughly corresponds to "Off". This can turn the service "off" *or* "on":

```
networksetup -setnetworkserviceenabled "Bluetooth" off
```

If you run this while the Network Pane is open in the GUI, the GUI will alert you with, "Your network settings have been changed by another application" – just something to be aware of. It then places an asterisk next to the name of the interface when using "-listallnetworkservices" and "-listnetworkserviceorder".

Finally, there are some very useful Airport specific commands. Different than enabling or disabling the interface, you can also turn power to the Airport off completely:

```
networksetup -setairportpower off
```



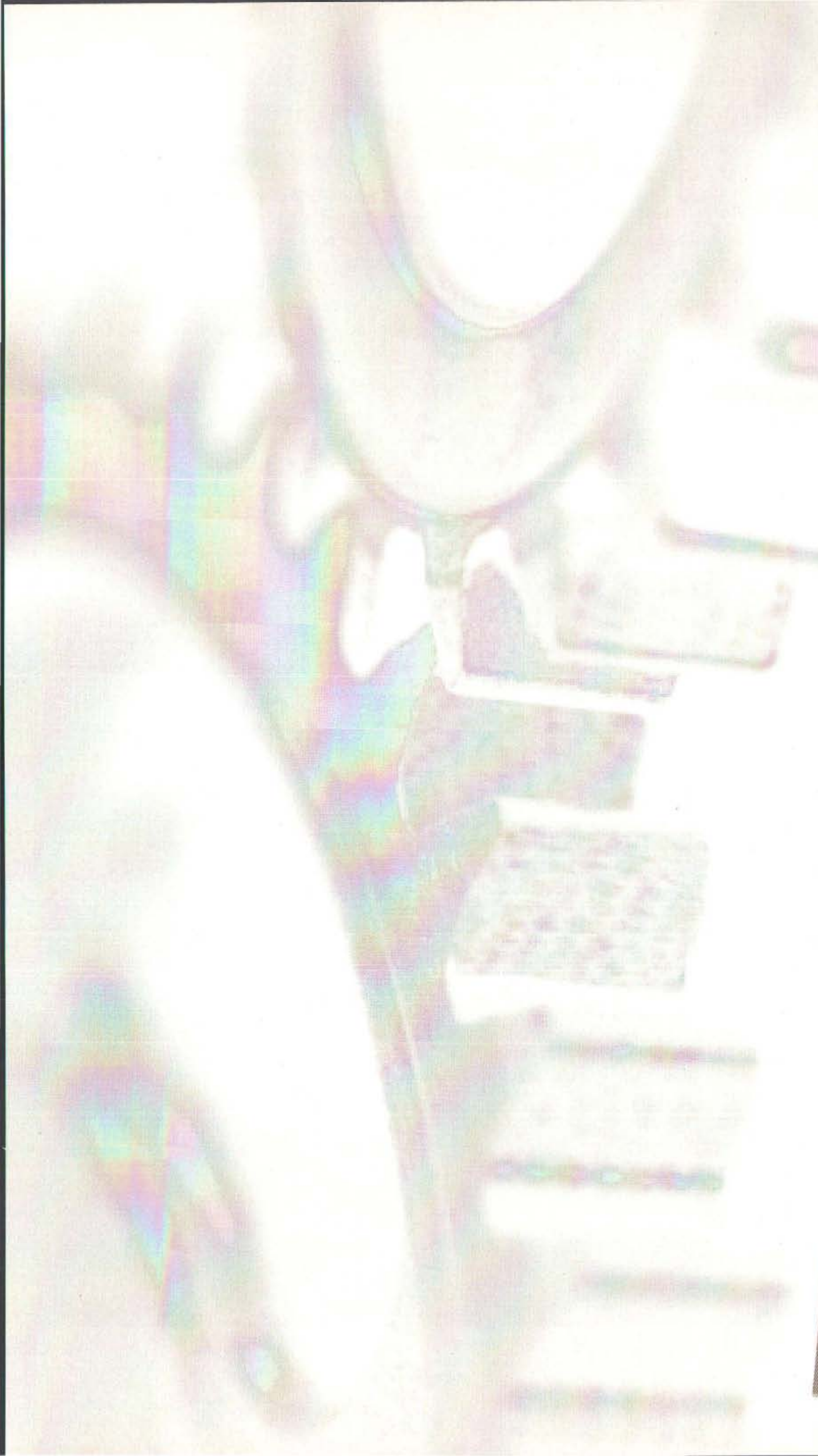
...Brings The Right Updates to You

VersionTracker Pro brings you everything you need to keep your software up-to-date, by monitoring all your installed software and alerting you as soon as updates become available, with download links and all the information you need to make an informed decision.

- **Know instantly when updates become available:** VersionTracker Pro monitors all your installed software, and alerts you as soon as updates become available.
- **The information you need finds you:** VersionTracker Pro brings you reviews, ratings, system compatibility, features, bug fixes, and download links for each update.
- **Manage the information influx:** Customize VersionTracker Pro to only show you information on the installed software you care about. Group applications to fit your work style

$49.95 for a three-Mac license **$79.95 covers up to ten Macs and PCs**

www.versiontracker.com/subscribe/mactech



SALES.
INVENTORY.
PLANNING.
MANUFACTURING.
DISTRIBUTION.
PURCHASING.
SHIPPING.
PROJECTS.
FINANCIALS.
CRM.

Enterprise software, all working together.

All on a Mac.

www.OpenMFG.com

Of course, you can turn it back on with the same switch and a parameter of "on". You can also check the current state with "-getairportpower".

Of greater use, though, is the ability to programmatically set which network to connect to, using "-setairportnetwork":

```
networksetup -setairportnetwork "BigNet" SekurePa$$wd
```

Here too, you can find out the current state with a 'get' variant:

```
networksetup -getairportnetwork
Current AirPort Network: SuperAirNet
```

The **networksetup** command is very powerful and very thorough! In addition to the options I covered here, you can do *anything* that you can do in the GUI: create VLANs, turn AppleTalk on or off, set proxy configuration and more. It's a good command to be familiar with for scripting, remote setup and just to do things the "OS X way!"

scp

"**scp**" is the secure copy program. It's part of the ssh suite, which means that any remote machine you can connect to via ssh, you can also copy programs to and from. While there *is* a man page for this one, it doesn't have example usage. Also, I find that not enough people know about or use it – even those who use ssh on a regular basis. **scp** is simple, really. To copy a local file to a remote machine, use this:

```
scp local_file
my_id@server.example.com:/path/to/file/filename
```

You'll be asked for a password from the remote machine. Then, given permission on the remote, you'll see a progress meter that shows the file in transit. In my example, "my_id" is the id you use on the remote machine. (For those of you that may want to date yourself, unlike ssh, scp does *not* accept the "-l" (ell) switch to pass in your credentials – a switch that ssh supplies to make the transition from telnet easier!).

The colon character (":") separates the hostname from the path. This can be a relative path, too. The starting location is the remote id's home directory. Also, like ssh, if unspecified, your current id is supplied. So, if I wanted to copy a file to my home directory on my test box, I could simplify the command down to this:

```
scp local_file lycaeum.radiotope.com:
```

Do note the trailing colon character. Without it, you'll just make a local copy of the file – in this case, named "lycaeum.radiotope.com".

To copy *from* a remote machine, just reverse the order of the files, putting remote information first. To copy "filename" from the remote to your machine as "local_file", try this:

iListen

Speech Recognition for Mac OS X

It's like having a third hand...

Your voice!

Customizable for any application!

Dictate **Command** **Control**

http://www.macspeech.com **(603) 350-0903**

```
scp my_id@remote.example.com:/path/to/file/filename
local_file
```

Again, to use the same name as the remote, simply omit it on the local side:

```
scp lycaeum.radiotope.com:program-4.7.6.tar.gz ./
```

If you're thinking of using this in an unattended script, you'll need to authenticate via *keys* so that you're not prompted for a password. While there are plenty of examples on the web of how to do this, for the sake of completeness, here is the short version:

Open up a shell on the local computer, and log into the correct local account.

Generate a public-private *keypair* by typing "ssh-keygen -t rsa". Leave the passphrase empty. This creates the files "id_rsa" and "id_rsa.pub" in the .ssh directory in this account's home directory.

Copy *only* id_rsa.pub to the remote machine using scp.

Login to the remote machine, and add the contents of id_rsa.pub to ~/.ssh/authorized_keys. You can use redirection to do this, as the authorized_keys file may already exist: cat id_rsa.pub >> ~/.ssh/authorized_keys

Logout of the remote machine, and test the setup using ssh. Try to ssh back into the remote machine. This time, you should not be prompted for a password.

Using keys in this manner, you'll be able to setup scp copies in an unattended script.

ssh and File Copy Programs

You may use ssh every day. You may use some of its more advanced features. But it is impossible to *Know* ssh. For every feature that you use, there seems to be another that you didn't know even existed. Did you know that ssh accepts input on standard in (stdin) and will simply shove it through the tunnel, popping it out on the other side?

What does that mean to us? Well, you can just gather up your data and pipe it to ssh:

```
tar czf - /path/to/file | ssh
```

Of course, we need to do something with it once it gets to the other side. How about expanding it somewhere? Let's imagine that we wanted to tar up the local /www directory and get it to a remote machine with its hierarchy intact:

```
tar czf - /www | ssh "cd /; tar xzfvp -"
```

Now, `tar` is a very nice solution – most of the time. It's had a bit of an on-again off-again brokenness to it under OS X regarding resource fork copying. As I write this article, using OS X 10.4.9 Intel, `tar` works very nicely for copying resource forks. One other huge advantage to `tar` is that it preserves dates on copy. But, in fact, `ditto` will do this too, along with preserving all of the other OS X-specific data that it normally does. A simple example is a webloc file, which typically is resource-fork *only* (text clippings are also resource fork only). Fire up Safari, load a page and then drag the icon from the

EazyDraw

Make Drawing Fun on OS X

Technical Drawings
Illustrations
Charts & Diagrams
Text Layout
Web Graphics
"This Advertisement"

EazyDraw is a well balanced 2D vector drawing application, not bloated with obscure computer illustration procedures. The new OS X user interface stays out of the way and lets you get your ideas on paper - or the web.

Import & Edit
MacDraw
ClarisDraw
PICT & PDF

http://www.EazyDraw.com

Phone: 608 444 5245
Fax: 608 635 2124

N 5040 Beach Garden Road
Poynette, Wisconsin, U.S. 53955

Resellers Call Jay: 608 469 8991

location bar to your desktop. Now you have a resource fork only file to work with:

```
$ cd Desktop

$ ls -l Weather\ Map.webloc
-rw-r--r--    1 marczak  marczak  0 Apr 15 08:33 Weather
Map.webloc

$ ls -l Weather\ Map.webloc/..namedfork/rsrc
-rw-r--r--    1 marczak  marczak  837 Apr 15 08:33 Weather
Map.webloc/..namedfork/rsrc
```

Let's move this file to a remote machine using **ditto**. Again, I'm going to copy to my test server (lycaeum) and rely on the fact that by default, ssh plops me into my home directory. If you want to write the data elsewhere, replace the "./" path with one that suits you (and you have permission for). Here it is:

ditto -c Weather\ Map.webloc - | ssh lycaeum.radiotope.com ditto -x - ./

Log into the remote server and check out the date, time and other file attributes – they'll all stay intact (however, not ACLs, as expected).

Machine Info

I'll leave you with two commands that can display information about the software and hardware on a given machine: sw_vers and system_profiler. Not that these are complex or need a lot of explanation, but just to know that they exist. If I could count all of the times I see a question about getting system information on a tech mailing list, I probably wouldn't be writing this!

sw_vers is the simpler of the two. With no options, it dumps out the product name, version and the build number:

```
$ sw_vers
ProductName:    Mac OS X Server
ProductVersion: 10.4.9
BuildVersion:   8P135
```

You can pass in switches to *limit* the amount of information returned:

```
$ sw_vers -buildVersion
8P135
```

Filtering the information is certainly useful if you need this information in a script.

system_profiler is the shell equivalent of the GUI System Profiler.app. Like sw_vers, just used on its own, it gives you a good deal of info. Try it! (There's way too much output to print here). You can parse through this data on your own. If you're only looking for a specific bit of info, you can pass in filters that do just this:

```
$ system_profiler SPMemoryDataType
```

Save time... save money...
synchronize your databases

***fmSQL Synch* eliminates double data-entry... forever.**

You can now synchronize your FileMaker and web (SQL) databases. Data is compared on a *record-by-record* as well as *field-by-field* basis. Existing databases can be configured for synchronizing with just a few modifications.

Some of the features include:

- ✓ Eliminates double data-entry
- ✓ Two-way synchronizing
- ✓ Field level conflict checking and resolution
- ✓ Setup Assistant
- ✓ Creates all SQL statements for you automatically
- ✓ Easily integrates with existing FileMaker solutions

Download a demo, and get more information at
www.fmSQL-Synch.com

Garrison Computer Services
info@garrison.com.au Ph: +61 2 4575-5247
www.garrison.com.au PO Box 141 Windsor NSW 2756 Australia

Your G5's Dream Date

The 3ware Sidecar & Your Apple Power Mac G5: The Perfect Match

The 3ware® Sidecar by AMCC is a powerful SATA RAID desktop storage solution designed specifically for the Apple® Power Mac® G5. At speeds 4-8x faster* than Firewire or USB, it can store and protect tons of your photos, songs, videos, illustrations and web pages.

Whether you're editing and archiving digital photo shoots or snapshots, home movies or future Academy Award® winning films, garage jam sessions or professional mixes — rest assured your data will always be protected. The 3ware Sidecar is designed for creative professionals and enthusiasts who care about their data and just can't risk losing it! With the 3ware Sidecar, your data is RAID protected, so a failed drive won't mean the loss of hours of creative output.

On the set, in your studio or at the office, the 3ware Sidecar lets you think outside the box. And with up to 3TB** of storage capacity, there's no need to worry about running out of space. Just install, set up and connect — it's as easy to use as 1-2-3.

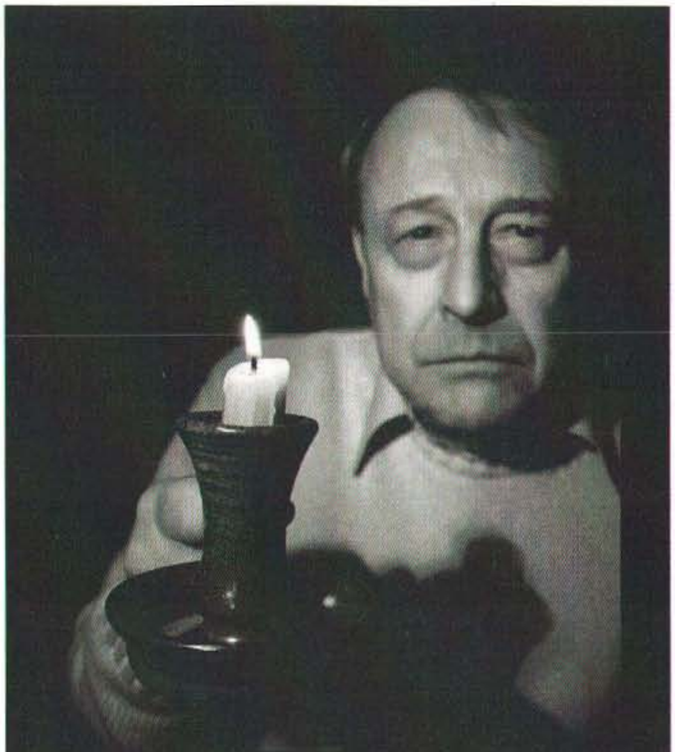
3ware Sidecar, the perfect partner for your G5 workstation — no dinner required.

AMCC STORAGE

Think Outside the Box
Find out more at **www.3ware.com**
Or call (877) 88-3ware; 877-883-9273

* SATA II: 300 MB/second; Firewire: 80 MB/s or 40 MB/s; USB 2.0: 48 MB/s
** Using four 750 MB SATA drives (not included)

Bell Microproducts	D&H	Synnex Corporation
1-888-394-2355	1-800-340-1001	1-800-756-9888



Computer down because the power is out again ?

UPS Power Battery backups from just

$25!

www.UPSforLESS.com
(866) 577-3628

UPS FOR LESS
experts in uninterruptible power supplies

```
Memory:

    BANK 0/DIMM0:

      Size: 2 GB
      Type: DDR2 SDRAM
      Speed: 667 MHz
      Status: OK

    BANK 1/DIMM1:

      Size: 1 GB
      Type: DDR2 SDRAM
      Speed: 667 MHz
      Status: OK
```

You can get a list of data types with the "-listdatatypes" switch. If you want more than one type at once, go ahead and pass in those types:

```
system_profiler SPDisplaysDataType SPAirPortDataType
SPPowerDataType
```

sw_vers and system_profiler are both great commands when accessing a remote machine that you may not be familiar with – especially when someone is sitting at the console and you don't want to ask them for the information!

Summary

I regularly use the commands and techniques presented here. While the information is typically in the man page, useful examples don't always accompany that information. Sometimes, you just need to experiment with a command – on a test system, of course – until you have it figured out. Sometimes there are alternatives to man. Many GNU utilities have an info page that's different than the man page, if one even exists. Try "`info emacs`", for example. Press 'q' to get out of `info`'s display. Of course, MacTech and publications like it (are there any?) present good alternatives to man pages as well. I hope this article was a good step in that direction for anyone reading it.

Media of the month: well, it's not going to be a Leopard title, as we're going to see a little delay there! However, there's plenty to be read before then. Don't get all stressed out about it. Instead, find some entertaining reading, like the "Lord of the Rings" trilogy. OK, not necessarily *short* reading, but I'm always surprised by how many people have *not* read these books. The movies were fine, but the books capture something different.

Until next month, enjoy!

MT

About The Author

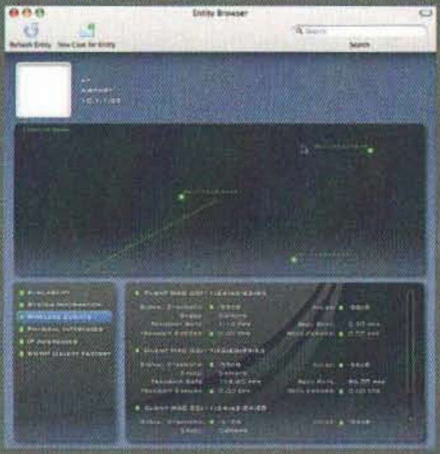
Ed Marczak owns Radiotope, a technology consulting company that brings enterprise solutions to small and medium sized businesses. He is also the Executive Editor for MacTech magazine. Outside of technology, Ed likes to spend time with his family, and to practice counting.

Network, Server and Appliance Monitoring.

For Mac OS X

Xserve RAID

Xserve (Intel & G5)


Airport

Lithium Network Monitoring Platform

Lithium can now monitor your Xserve, Xserve RAID, Qlogic switches, Airports, Mac OS X Server... and everything else in your network.

www.lithiumcorp.com

lithium Corp.

Perl 6 On XCode

Bringing the power of Perl 6 to the XCode environment

by José R.C. Cruz

Introduction

First developed in 1987 by Larry Wall et al, Perl has evolved into the scripting language of choice for developing software solutions such as CGI, system administration scripts, and database management tools. It is also one of the first scripting languages designed to be platform-agnostic. Furthermore, it comes with a very extensive library of functions and subroutines, which can also be extended or improved.

This article introduces the next major revision of the Perl language, version 6. It will provide a concise overview of the language, especially in areas where it improves upon its predecessor, Perl 5. It also introduces two open-source projects, Pugs and Parrot, both of which will become the basis of the new Perl 6 runtime system. It will then demonstrate how to use the XCode IDE to port these projects to OS X, as well as how to add support for Perl 6 to the IDE. A copy of this project can be obtained at the following URL: <ftp.mactech.com/src/mactech/volume23_2007/23.05.sit>.

Enter Perl 6

A Brief Background History

Perl 6 started out in 2001 as a series of 8 documents known as the *Apocalypses*. These documents address the current shortcomings of the Perl language, as well as the proposed changes and additions to improve it. Each Apocalypse document is numbered to correspond to a specific chapter from the official Perl book, *Programming in Perl*, also fondly referred to as "The Camel Book".

A second set of 6 documents, collectively known as the *Exegeses*, provides a more detailed description of the changes proposed in the Apocalypses. They also provide a number of code examples to illustrate the new syntax and semantics of the Perl 6 language. Later on, a third set of 12 documents, the *Synopses*, contains the official design specifications of the entire Perl 6 environment. For a more detailed look at these documents, visit the official Perl 6 home page at <http://dev.perl.org/perl6>.

The fact that the development of Perl 6 started with design specifications is unique in the history of the language. Earlier versions of Perl were developed without the benefit of a single specifications document. In fact, each time discrepancies were found between the official user's documentation and sources, it is often the former that gets updated. Compare this with other modern languages such as FORTRAN or C, whose compilers and interpreters are expected to comply with their respective standards.

The Perl 6 Advantage

Perl 6 is essentially a complete overhaul of the entire Perl language. When completed, it will bring a large number of long-awaited features and improvements to the Perl development community. Some of the more notable ones are as follows:

- support for static data types
- formalized and consistent parameter lists
- strict sigil invariance
- more robust object oriented design
- improved regex support
- simplified and streamlined grammatical syntax
- support for preprocessor macros
- an actual switch statement

Explaining each Perl 6 feature in great detail is beyond the scope of this article. Furthermore, the specification is still a work-in-progress. Those who are interested in learning more about the language, as well as how to start writing Perl 6 scripts, should consult the book, *Perl 6 Essentials*, which is listed at the end of this article.

A note to all Perl users: The Perl 6 specification is not designed to be backward compatible to previous versions of Perl. However, it does support the creation of a compatibility layer that will allow the execution of Perl 5.x scripts. The only limitation is that these scripts will be unable to avail themselves of the new language features.

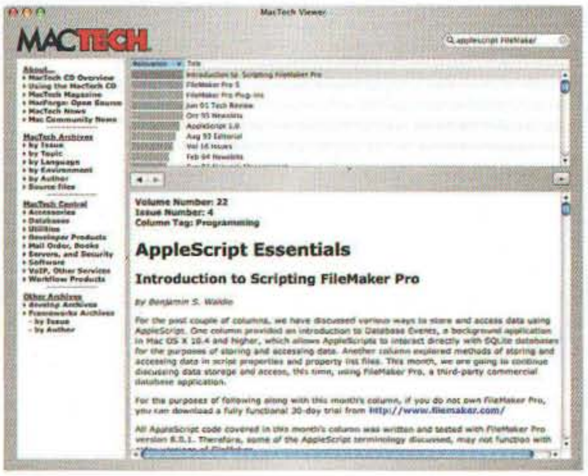
Missing a few issues?

Here is your chance to catch up.

The MacTech CD - Volumes 1.01-22.09 is packed with more than ever before -- The entire archive of MacTech Magazine, that's over 2700 articles from more than 250 issues (1984 - September 2006), all 29 issues of Apple's develop, 21 issues of FrameWorks magazine, all of the source code, working applications, full documentation, demos for techs, and more!

Everything is displayed in the all new, very fast, very searchable ***MacTech Viewer!*** A new application that has been designed specifically with Techs in mind. Search quickly through *20 plus years* of great information provided by MacTech. Information to make your life easier.

Requires Mac OS X v. 10.4.5 or later



Volume Number: 22
Issue Number: 4
Column Tag: Programming

AppleScript Essentials

Introduction to Scripting FileMaker Pro

by Benjamin S. Waldie

For the past couple of columns, we have discussed various ways to store and access data using AppleScript. One column provided an introduction to Database Events, a background application in Mac OS X 10.4 and higher, which allows AppleScripts to interact directly with SQLite databases for the purposes of storing and accessing data. Another column explored methods of storing and accessing data in script properties and property list files. This month, we are going to continue discussing data storage and access, this time, using FileMaker Pro, a third-party commercial database application.

For the purposes of following along with this month's column, if you do not own FileMaker Pro, you can download a fully functional 30-day trial from http://www.filemaker.com/

All AppleScript code covered in this month's column was written and tested with FileMaker Pro version 8.0.1. Therefore, some of the AppleScript terminology discussed, may not function with

Toll Free 877-MACTECH, Outside US/Canada: 805-494-9797 • http://www.mactech.com/cd/

The Perl 6 Projects

Two open-source projects are currently being developed to implement the Perl 6 specification. The first project, the **Perl 6 User's Golfing System** (Pugs), attempts to implement the Perl 6 language specification as defined in the Synopses. Its end products are two command-line tools: `pugs`, an interpreter with an interactive shell, and `pugscc`, a Perl 6 compiler. The latter tool is capable of compiling an existing Perl 6 script into Perl 5, JavaScript, or `PIR` (more on this later) for execution.

At the time of this article, the latest stable version of Pugs is 6.4.11. The source tarball for this project, together with instructions on how to compile it, can be downloaded at <http://www.pugscode.org>.

Unlike most open-source projects, however, Pugs is written in Haskell, which is a pure functional programming language with non-strict semantics. Since GCC does not have built-in support for Haskell, the **Glasgow Haskell Compiler** (GHC) needs to be installed into OS X in order to compile the Pugs sources. An OS X installer for the latest version of GHC (6.4.1), which also fixes a compatibility bug with GCC 4.0, can be downloaded at <http://haskell.org/ghc/download_ghc_641.html#macosx>. Furthermore, the `hs-plugins` library needs to be installed with GHC. This library enables GHC to compile, load, and evaluate Haskell sources at runtime. The OS X installer for this library is available at http://darcs.net/DarcsWiki/ ¬
CategoryBinaries#headc14449f675f1f36c70703538d40e79f056a69bd9

The second open-source project is **Parrot**. This project attempts to create a register-based software CPU that is designed to run a wide range of languages as efficiently as possible. It uses a multi-layered design that enables it to support both dynamic languages such as Perl or Ruby, as well as static ones such as Java. It also uses just-in-time compilation to improve its execution speed. Once completed, it will become the de facto runtime engine for Perl 6.

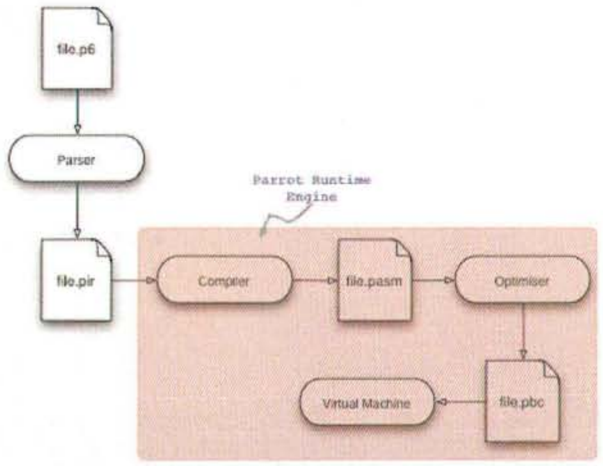


Figure 1. The Parrot Runtime Engine.

Figure 1 shows a simplified diagram of the Parrot engine. Here, a parser (usually Pugs) converts a Perl 6 script into an intermediate file format known as a PIR, or **Parrot Intermediate Representation**. This PIR file is then compiled into assembly code, which is then optimized and translated into byte code for execution by the virtual machine.

At the time of thiswriting, the latest stable version of the Parrot engine is version 0.4.4. This version contains an initial implementation of a Perl 6 compiler, as well as compilers for APL and Scheme. It adds support for hierarchical class names, tree transformations, and rules/operators precedence. It also has an improved and redesigned grammar engine.

The latest Parrot source tarball, together with instructions on how to compile the project, is available at the Parrot home page located at http://www.parrotcode.org/source.html
Since Parrot requires at least GCC 3.x and Perl 5.6 for compilation, it will require version 10.3 or newer of MacOS X.

Porting Perl 6 with XCode

The XCode Advantage

Like all open-source projects, compiling Parrot and Pugs is essentially a command-line affair. Both projects use the venerable makefile system to configure and control their respective compilation process.

However, with a little work, the XCode development environment can be used to port these open-source projects to OS X. Its user-friendly text editor, together with its well-designed search/replace features, can be quite useful in navigating the plethora of files and scripts that make up these projects. Also, the configuration, compilation, and even the installation processes for these projects can be controlled to a fine degree through the judicious use of the Run Script build phase.

Parrot in XCode

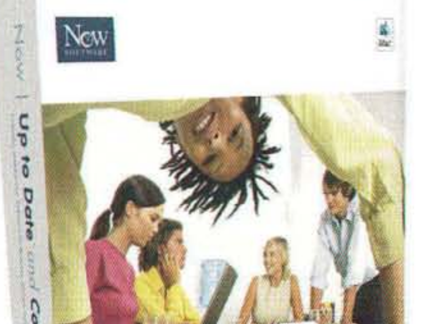
The XCode project for Parrot (Figure 2) is created using the Empty Project template from the New Project Assistant dialog. Its project directory contains all the files and subdirectories obtained from the Parrot source tarball. The project is assigned with three build configurations: Debug, Release, and Install. The first two are standard configurations provided by the project template, while the last one is used by XCode to install the resulting binaries and support files directly into the host OS X system after a successful compile and test run.

The Parrot project has two targets: Parrot Make, which is the default target, and Parrot Clean. The Parrot Clean target is an external target. Its sole function is to reset the entire project to an uncompiled state when the Clean option is chosen from the Build menu. The Parrot Make target, on the other hand, is a shell script target. It contains four Run Script phases, each phase controlling an aspect of the Parrot build process.

Now **scheduling** and **contact management** for your entire organization.

Now | **Up to Date**® and **Contact**® **5**

Calendar and Contact Software for Business, Groups, Families and You.


Now
Now | Up to Date and Contact 5
Now | Up to Date and Contact 5

Is this project on schedule? When are you available to meet about the systems upgrade? Where are all the field techs today? When was the last time anyone talked to our biggest customer?

Virtually all groups live (or die) by their abilities to meet deadlines and keep track of their customers, prospects, and vendors. Few small companies or even departments of big companies have the tools they need.

Now Up-to-Date & Contact might just be the calendar and contact software for you. It's time-tested and used by more Mac-based companies than any other solution. And it's cross-platform—available for your PC users, too. It's easy to install and manage and simple for your employees to understand and use.

Using Now Up-to-Date & Contact you can schedule meetings for multiple users, view multiple, simultaneous calendars, and reserve rooms and resources. You can share contact information about your customers, prospects and vendors. And using our free server software you can set it up in minutes and share with users in the office or from anywhere with an internet connection.

Call us now at 866-527-0556 or email us at mactech@nowsoftware.com and we'll send you our free evaluation kit, including the book that will make it all easy, "Take Control of Now Up-to-Date & Contact" from Take Control books!

NOW SOFTWARE

Phone: 866-527-0556

Web: www.nowsoftware.com

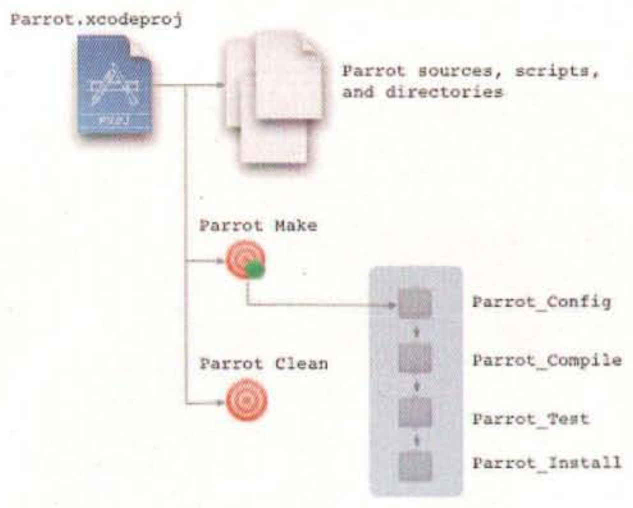


Figure 2. The Parrot XCode project.

The Parrot_Config phase first checks the current build configuration by querying the `CONFIGURATION` setting. It then executes the Perl script, `Configure.pl`, while passing the desired installation directory using the `--prefix` option. Once the script has finished, it then checks to see if any errors occurred during configuration, and if at least four of the principal files were generated.

The second phase, Parrot_Compile, starts the compilation process by invoking the `make` tool. The tool then uses the `Makefile` generated during the configuration phase. Again, the phase checks for any errors generated during the process.

The next phase, Parrot_Test, runs a series of standardized tests to see if the newly built Parrot engine is working correctly. These test scripts are bundled with the Parrot sources and should not be altered to ensure reliable results.

This phase first queries the `PARROT_TEST_MODE` setting to determine the type of test that needs to be run. If the setting is set to 1, the test process is started by invoking the `make` tool with a `test` option. A setting of 2, on the other hand, will invoke the tool with a `fulltest` option. Any other setting will cause this phase to skip the test process entirely.

The fourth and final phase, Parrot_Install, controls the installation of Parrot binaries and support files. Available only to the Release and Install configurations, it starts the installation process by invoking the `make` tool with an `install` option. The Debug configuration, however, skips this phase entirely, and an unknown configuration will generate an error signal.

The destination directory used by the installation process is the one set in the Parrot_Config phase through the `--prefix` option. For the Release configuration, the process uses the directory specified by the build setting, `INSTALL_PATH`. By default, this is set to the value of

Mail to FileMaker Importer

Automatically import email data from Mac OS X Mail or Microsoft Entourage into a FileMaker Pro database.

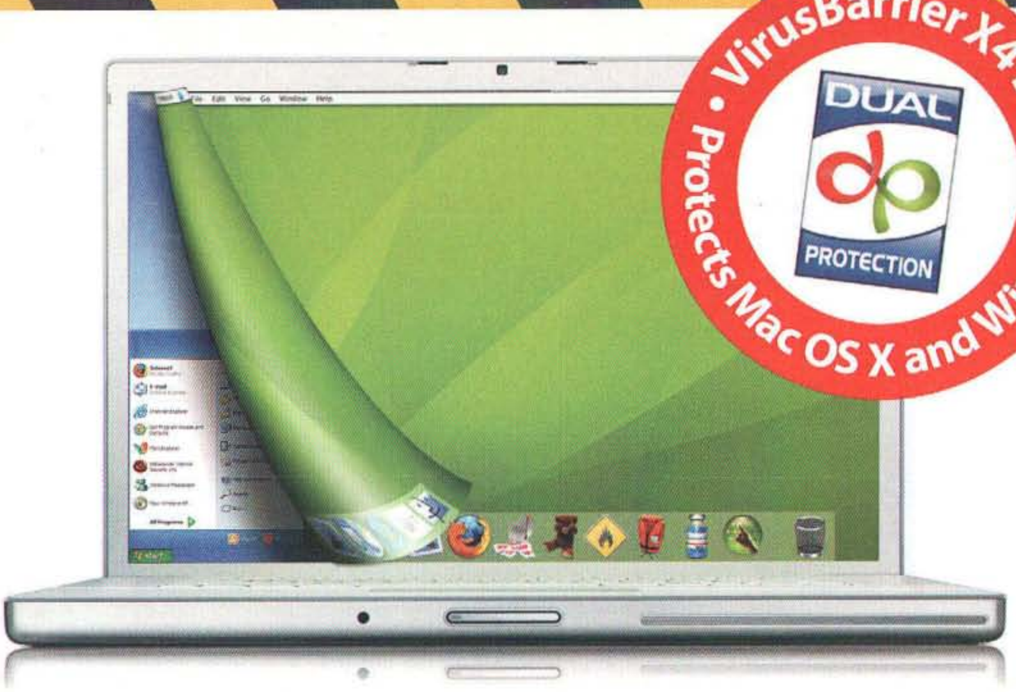
Automator Actions

- InDesign
- Photoshop
- Quark
- Finder
- iPhoto
- System
- More!

AUTOMATED WORKFLOWS

AppleScript • Automator • Workflow Automation Products and Custom Development Services

www.automatedworkflows.com



The #1 Mac antivirus solution and the #1 Windows antivirus program in a single package

Running Windows on a Mac is possible with today's Intel-based Macs, but this opens up a whole new range of security threats. Viruses, spyware, adware and hackers are all waiting to infect your Windows installation.

Intego, the leader in Mac security software, and BitDefender, the top PC security vendor, have joined forces to offer you **Intego Dual Protection**.

Protect Mac OS X and Windows from all known viruses with **Intego VirusBarrier X4 DP***.

Keep Mac OS X and Windows safe from hackers, vandals, viruses, spyware, spam and phishing with **Internet Security Barrier X4 Antispam Edition DP****.

Now, more than ever, you need full protection for both Mac OS X and Windows. Intego Dual Protection gives you all the security you need.

#1 Virus Barrier + #1 bitdefender

VirusBarrier rated best Mac antivirus by Macworld - BitDefender rated best Windows antivirus by PC World

* Contains Intego VirusBarrier X4 and BitDefender Antivirus.

** Contains Intego NetBarrier X4, VirusBarrier X4, Personal Antispam X4, and BitDefender Antivirus, Firewall, Antispyware, Antispam and Antiphishing.

DUAL dp PROTECTION

www.intego.com

intego

```
$(BUILD_DIR)/Library/Developer/Perl6/parrot
```

where **BUILD_DIR** is the location of the **build** directory with respect to the **Parrot.xcodeproj** bundle. The Install configuration, on the other hand, uses the one specified by **PARROT_PATH_INSTALL**. This setting default to

```
$(LOCAL_DEVELOPER_DIR)/Perl6/parrot
```

where **LOCAL_DEVELOPER_DIR** is the **/Library/Developer** directory of the OS X boot volume. In retrospect, this directory proved to be a more appropriate place to install additional developer tools. Most open-source projects usually install their end products in either **/user/local** or **/opt/local**. However, unlike those two directories, **LOCAL_DEVELOPER_DIR** is a public directory and, as such, does not require any **sudo** privileges to be accessed.

Pugs in XCode

The second XCode project, Perl6 (Figure 3), is also created using the Empty Project template. Its project directory contains all the files and subdirectories that came with the Pugs tarball file. Furthermore, it contains a reference link to the **Parrot.xcodeproj** bundle. It also shares the same three build configurations as the Parrot XCode project.

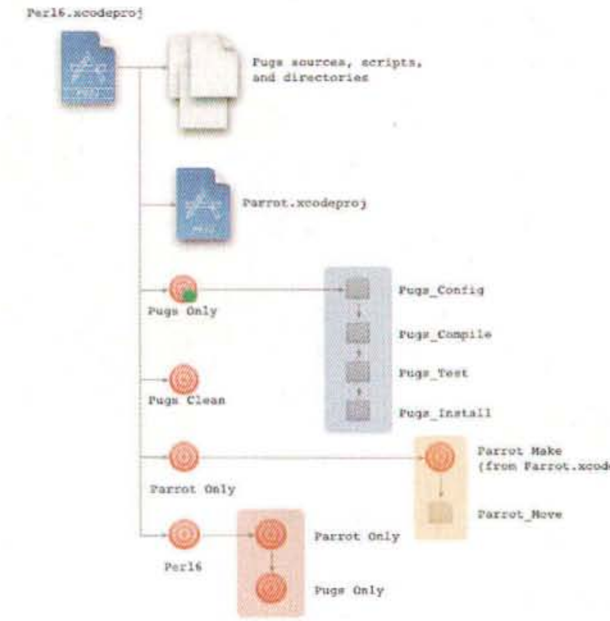
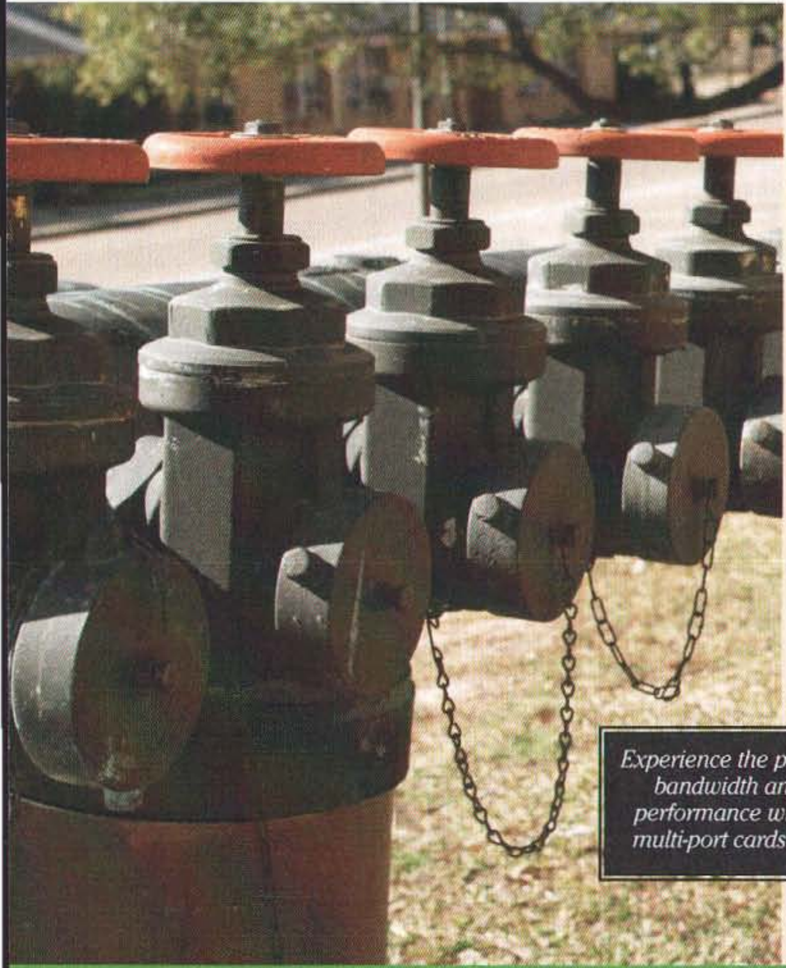


Figure 3. The Perl 6 XCode project.

The Perl6 project has four targets. Parrot Only is an aggregate target that directly references Parrot Make from the Parrot XCode project. Selecting this target will start the



IMPROVE YOUR WORKFLOW

IF YOU'RE READY TO TURN UP THE FLOW OF YOUR NETWORK, WE'VE GOT THE TOOLS TO MAKE IT HAPPEN. WE'RE THE MAC NETWORKING EXPERTS WITH ROCK SOLID SOLUTIONS AND TOP QUALITY PRODUCTS. LIKE OUR NEW EDGE-CORE® SWITCHES THAT DELIVERS ENTERPRISE QUALITY AT A FRACTION OF THE COST. SO IF YOU'RE READY TO CRANK UP YOUR NETWORK PERFORMANCE, CALL US. WE'RE HERE TO HELP.

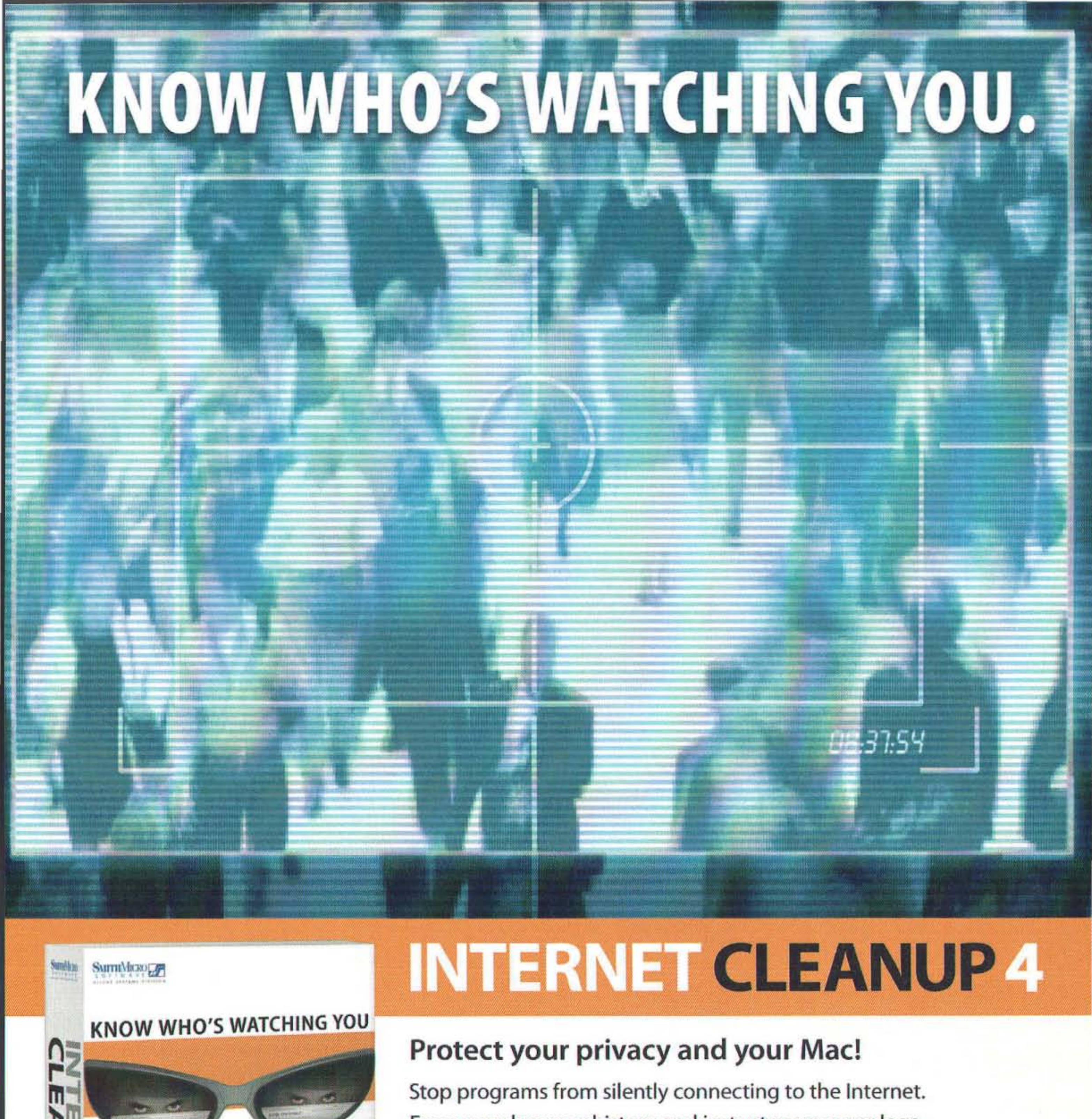
Experience the power of greater bandwidth and improved performance with Small Tree multi-port cards and switches.

SmallTree
COMMUNICATIONS

www.small-tree.com • 866.STC.4MAC

Edge-corE
Powered by Accton

Now available in the U.S. and Canada exclusively from Small Tree - the Mac networking experts



INTERNET CLEANUP 4

Protect your privacy and your Mac!

Stop programs from silently connecting to the Internet.
Erase your browser history and instant messenger logs.
Search and filter your email attachments.
Prevent others from invading your files.
Permanently shred your private files.
Block pop-ups and Flash ads.

Mac Universal

Fight back and safeguard your Mac.
DOWNLOAD YOUR FREE TRIAL TODAY!
www.allume.com/mac/cleanup/

SMITHMICRO SOFTWARE
ALLUME SYSTEMS DIVISION
WWW.ALLUME.COM

build process for that project. The target also contains a single Run Script phase, Parrot_Move. The function of this phase is to copy the resulting Parrot binaries and files into the Perl 6 build directory if the current build configuration is set to Release. Why use a Run Script phase as oppose to a Copy Build phase? The Copy Build phase does not provide any means of detecting the current build configuration. Only the Run Script phase has this capability by checking the CONFIGURATION build setting.

The second target, Pugs Clean, is an external target that is used by the project to reset itself back to an uncompiled state. It also invokes the Parrot_Clean target from the Parrot XCode project. Perl 6 is another aggregate target that invokes the two targets, Parrot Only and Pugs Only, in their respective order. This target is used when building and testing both the Parrot engine and the Pugs tools.

The fourth and final target is Pugs Only. Like Parrot Make, this shell script target has four run script phases. The first phase, Pugs_Config, starts the configuration process by running the Perl script, `Makefile.pl`. It also sets the installation directory for the final product by passing the directory path using the `PREFIX` option. It then checks for any configuration errors, and to see if a valid `Makefile` has been created.

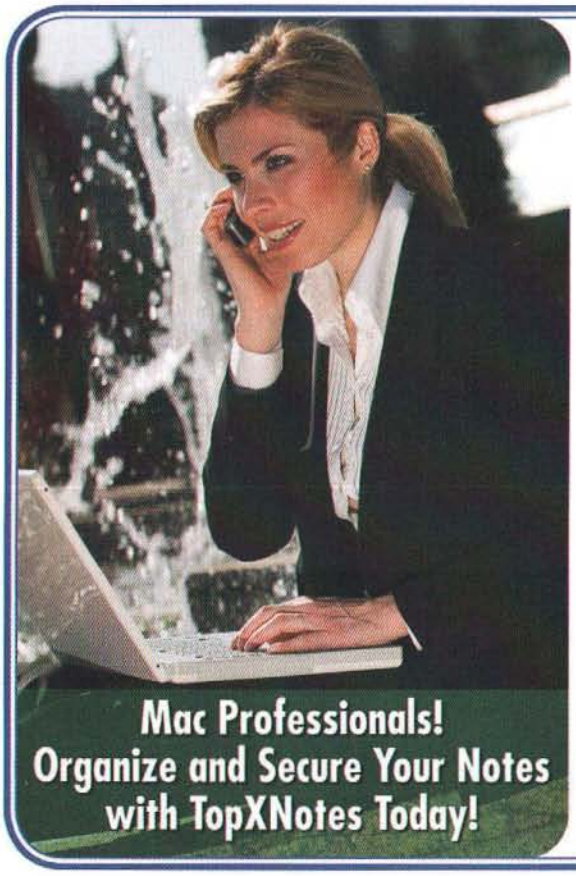
The second phase, Parrot_Compile, first checks the build setting, `PUGS_OPTIMISED`, to determine how to compile the Pugs tool. A setting of 1 will invoke the make tool without any command options. Any other setting will invoke the same tool with an `unoptimized` option.

The choice of compilation option directly determines the runtime performance of the Pugs tools, as well as the length of the compilation process. An optimized compilation takes a considerable amount of time to complete, but results in binaries that provide fast Perl 6 interpretation and/or compilation. Conversely, an unoptimized compilation has a much shorter completion time. However, it results in binaries with much slower runtime performance. Overall, choosing the appropriate option will depend largely on how the final product is used or distributed.

The third phase, Pugs_Test, runs a series of standard test scripts to ensure that the resulting Pugs tools will perform as designed. It starts the test process by invoking the `make` tool with a `test` option. Since Pugs requires the Parrot engine in order to process Perl 6 scripts, this phase will first check to see if the latter is installed at the directory path specified by `PARROT_PATH_EXE`. Only then will it proceed with the test process.

Be aware that the Pugs test process takes a long time to complete. Unless the host system has enough computing horsepower to spare, this process should either be disabled for debugging purposes, or assigned to a separate system for execution. To disable the test process, set the build setting, `PUGS_TEST_MODE`, to 0.

The fourth and final phase, Pugs_Install, prepares the installation process for the current build configuration. As with Parrot_Install, this phase is available only for the Release and Install configurations. It is skipped in its entirety by the Debug configuration. Any other configurations, however, will generate an error signal.

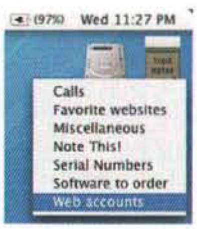


Try TopXNotes!

A modern notepad for Mac OS X loaded with handy features!

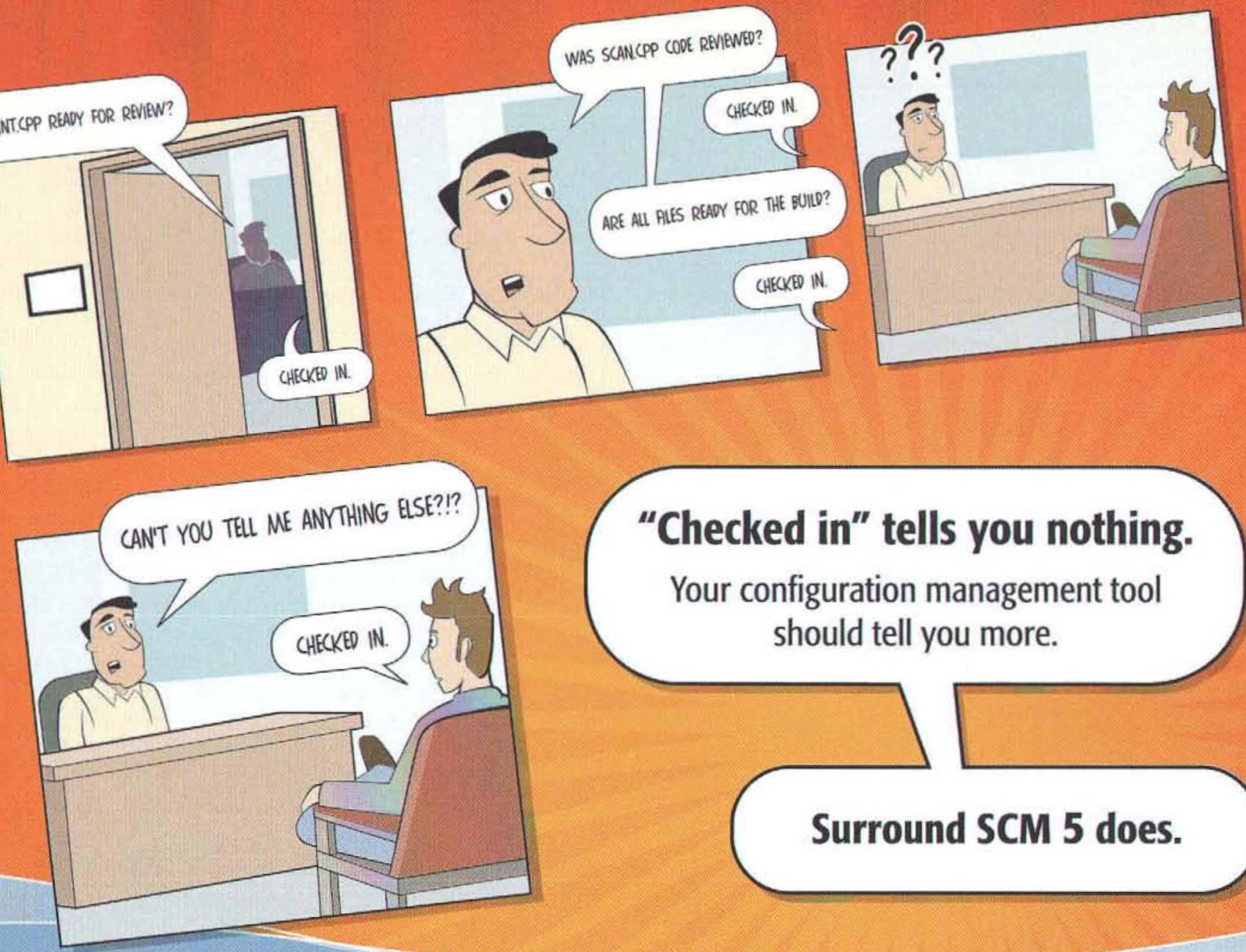
✓ QuickNotes - instant access
✓ MultiView - multiple note views
✓ Encryption - keeps sensitive data safe
✓ Automatic link recognition
✓ Drag/drop, and much more…

✓ Try **TopXNotes**
✓ iPod Users - try **TopXNotes iPod** to organize your notes and export to your iPod



Tropical Software
www.topxnotes.com

MacUpdate 5.0 Rating
versiontracker 5.0 Rating


IS PRINT.CPP READY FOR REVIEW?
CHECKED IN.
WAS SCAN.CPP CODE REVIEWED?
CHECKED IN.
ARE ALL FILES READY FOR THE BUILD?
CHECKED IN.
???
CAN'T YOU TELL ME ANYTHING ELSE?!?
CHECKED IN.

"Checked in" tells you nothing.

Your configuration management tool should tell you more.

Surround SCM 5 does.

Seapine

Surround SCM® 5

Software Configuration Management

Surround SCM 5 with change automation and custom metadata gives you more control over your software change process and a clearer view of the state of files.

Know whether a file you are including in the build has been code reviewed. Ensure design documents have been through the review process. Control who can make changes to files after they have been approved.

Surround SCM 5's virtual branches and collaboration features help streamline parallel development. There's never been a better time to change to Surround SCM—the next evolution in SCM.

- Create workflows to ensure changes to files follow your processes.
- See beyond "Checked in" to know the true status of files in the change process.
- Automate the development process with powerful in-application triggers.
- Improve security with single sign-on support.
- Integrate seamlessly with TestTrack and other popular development and build tools.
- Streamline communication with automated change notifications and flexible reporting.

Seapine ALM Solutions:

TestTrack Pro
Issue & Defect Management

TestTrack TCM
Test Case Planning & Tracking

Surround SCM
Configuration Management

QA Wizard Pro
Automated Functional Testing

Download your **FREE** fully functional evaluation software now at **www.seapine.com/mtsscm5** or call 1-888-683-6456.

Seapine Software™

SyncDeK®

Any Time,
Any Where Data

- **Mobility**: Data delivered any time, anywhere in the world.
- **Synchronize data** between FileMaker databases and now SQL data sources, easily and securely, no matter where they are.
- For offline laptops or online home offices and multiple office locations – each local copy of the database synchronizes its data with the others, **automatically or on demand**.
- SyncDeK **encrypts** all data transmissions.
- SyncDeK works with your databases behind your firewall to **prevent intrusion**, or keeps all database copies completely decentralized.
- Protect mission critical data with SyncDeK's continuous offsite differential **data backup**, warm standby and fail-over servers.
- **Sync servers** with client-free server replication.
- **Version Update Manager** allows you to update all copies of the database without requiring installation or any action on their part.
- **Call for demo, trial and best pricing options.**

WorldSync

877.548.4920 toll-free
510.548.4920 international
www.worldsync.com

The Release configuration uses the path specified by the build setting, **INSTALL_PATH**, to install the resulting Pugs tools and support files. This setting has the default value of

```
$(BUILD_DIR)/Library/Developer/Perl6
```

The Install configuration, on the other hand, uses the one specified by the build setting, **PUGS_PATH_INSTALL**. This one has the default value of

```
$(LOCAL_DEVELOPER_DIR)/Perl6
```

Integrating Perl 6 with XCode

The First Perl 6 Script

Once the Perl 6 binaries and support files are installed into the host OS X system using the previous two XCode projects, you can start writing your Perl 6 scripts. If you prefer executing your scripts through the Terminal prompt, make sure to first update the **PATH** variable by inserting the entries shown in Listing 1 to your `.bash_profile` file. Use your favorite text editor to update this invisible file, and make sure to restart your Terminal session in order for the changes to take effect.

Listing 1. Updating the .bash_profile file.

```
#define the Perl6 tools path
#
PERL6_ENV=/Library/Developer/Perl6

#start of PARROT configuration
PERL6_PARROT=${PERL6_ENV}/parrot
export PUGS_EMBED=perl5
PATH=${PERL6_PARROT}/bin:${PERL6_PARROT}/lib:${PATH}
#..end of PARROT configuration

#start of PUGS configuration
PATH=${PERL6_ENV}/usr/bin:${PERL6_ENV}/lib:${PATH}
#..end of PUGS configuration
```

Then write the script (let us call it **hello.p6**) using a text editor. A typical script should have at least the following entries.

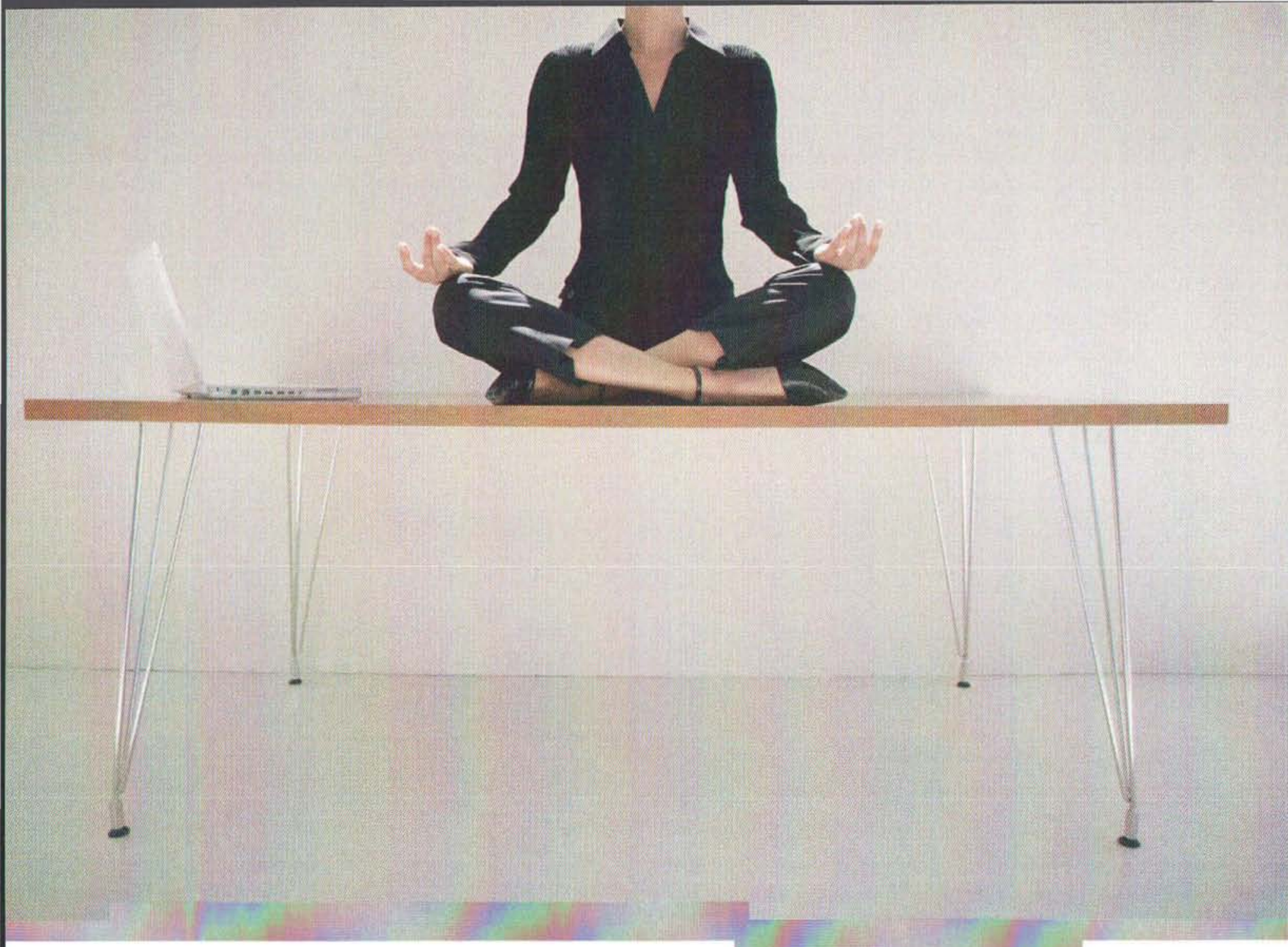
```
use v6.0;
say "Hello there, and welcome to Perl 6!"
```

Notice that the **use** keyword is used to indicate that the script is to be interpreted or compiled as a Perl 6 script. Then type **pugs hello.p6** at the Terminal prompt to run the Perl 6 script. If the installation was successful, and all Perl 6 binaries and support files are properly compiled, Pugs should print the following output to the Terminal window

```
Hello there, and welcome to Perl 6!
```

The Perl 6 Project Template

However, you can use XCode to write and execute your Perl 6 scripts without having to start a Terminal session. This approach allows you to make use of the excellent features provided by XCode, such as search and replace, and integrated source-code management. You can also add a standard project template to XCode in order to maintain consistency between projects as well as maximize code reuse.



out of your head...

REALbasic

Into your Mac

Would you like to create your own Mac software...maybe a business application, a graphics utility, or a game?

REALbasic 2007 is the fastest, easiest way to make software for the Mac. It is a modern, object-oriented language and environment that creates native software for Mac OS X, Windows and Linux from one set of source code.

Build a Universal Binary with one mouse-click!

www.realbasic2007.com

Figure 4 shows the contents of a basic Perl 6 project template. This template is based on the Empty Project template and it contains a single source file named **main.p6**. It also contains a reference to the Perl 6 library files, which should be installed in the directory **/Library/Perl6**. It has a single Shell Script Target named Perl6 Compile, which itself has a single Run Script phase named Run Perl6.

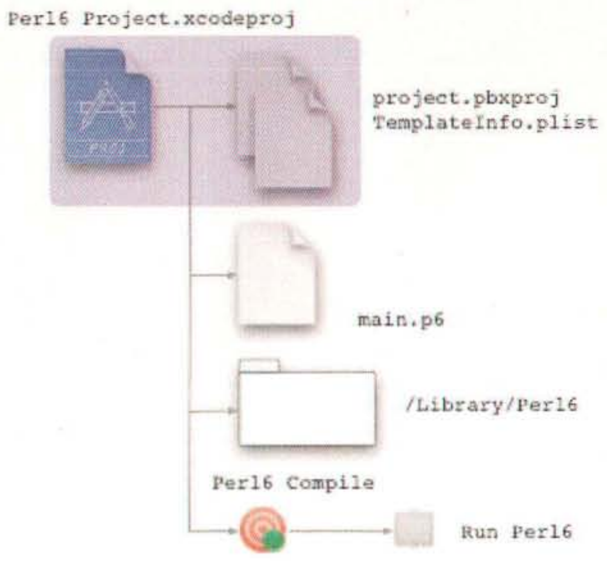


Figure 4. The Perl 6 project template.

Listing 2 shows the basic contents of the **main.p6** file. The file starts of with a standard header comment written using POD tokens. POD, or **Plain Old Documentation**, is markup language used for writing documentation for Perl scripts and modules. It serves the same purpose as HeaderDoc and JavaDoc.

Following the header comments is the Perl keyword, **use**. The numerical value after this keyword indicates that the entire script itself is a valid Perl 6 script and should be treated as such. A value less than 6 (e.g. 5.8.6) will tell Pugs to run the script in its Perl 5 compatibility layer.

Finally, right after that keyword is the first Perl 6 keyword, say. This will print out the string argument to **stdout** and append a new line to the output. It works similarly to the traditional Perl 5 statement of

```
print "Hello there, and welcome to Perl 6!\n";
```

Listing 2. The main.p6 file.

```
#!/Library/Developer/usr/bin/pugs
# Description of the file.
#
=pod
=head1 Project
    «Name_of_the_project»
=head2 Created by:
    «Name_of_the_software_developer»
=head2 Created on:
    «Date_when_the_project_was started»
=head2 Copyright:
    «Copyright_Year» «Name_of_the_organisation». All rights
reserved.
=cut

# basic Perl 6 statement
use v6.0;
say "Hello there, and welcome to Perl 6!";
```

Figure 5 shows the project settings used by the Perl 6 project template. The setting, **PERL6_PATH**, points to the Perl 6 directory where Pugs and Parrot are installed. The **PATH**

A BEST-IN-CLASS
NETWORK
IS WITHIN YOUR BUDGET

Cisco® • Juniper® • Extreme®

Does a tight budget compromise your network quality, functionality and reliability? With guaranteed pre-owned networking equipment from Network Hardware Resale, your network runs on top-tier Cisco equipment that performs like new, at a fraction of the cost. Equipment is delivered overnight with your configurations pre-loaded. Unparalleled warranty and support service leave nothing to chance. That's best-in-class.

To learn more, email nhrsales@networkhardware.com or call 805-690-3754.

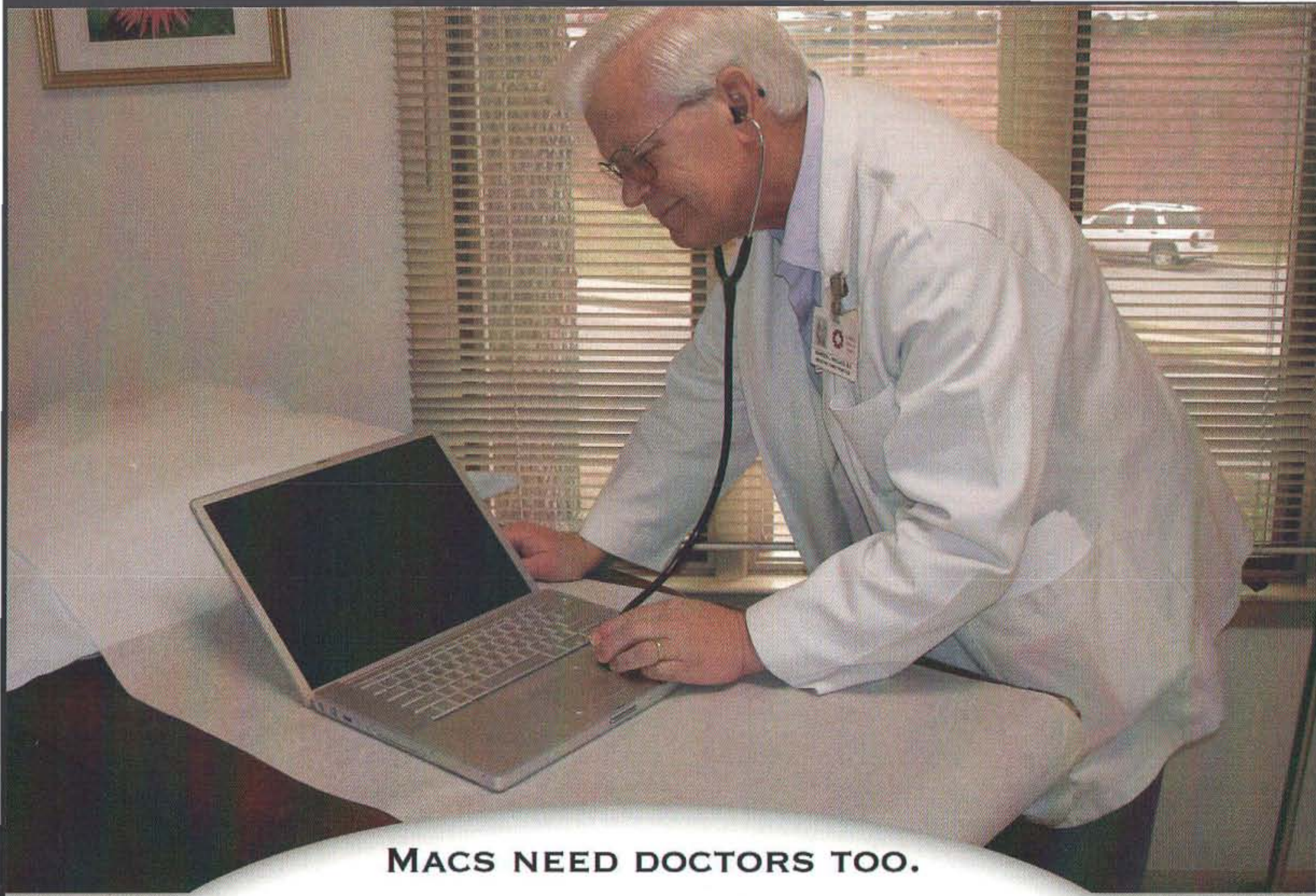
One-year warranty available to retail customers only. Overnight delivery subject to FedEx service restrictions. Network Hardware Resale and the Network Hardware Resale logo are trademarks of Network Hardware Resale LLC. All other company or product names are property of their respective owners.
© 2006 Network Hardware Resale LLC.

www.networkhardware.com

Routers • Switches • Security
Access Servers • VoIP • Memory

- **Buy, sell** or **trade** pre-owned equipment
- **One-year, advanced replacement** warranty
- Prices **50% to 90% off** list
- **Overnight** delivery
- Cisco certified **technical support** available
- More than **$100 million in inventory – current and past generation** products

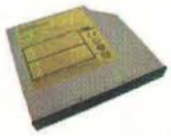
NETWORK HARDWARE RESALE
The World's Most Trusted Source™



MACS NEED DOCTORS TOO.

Powerbook Medic .com

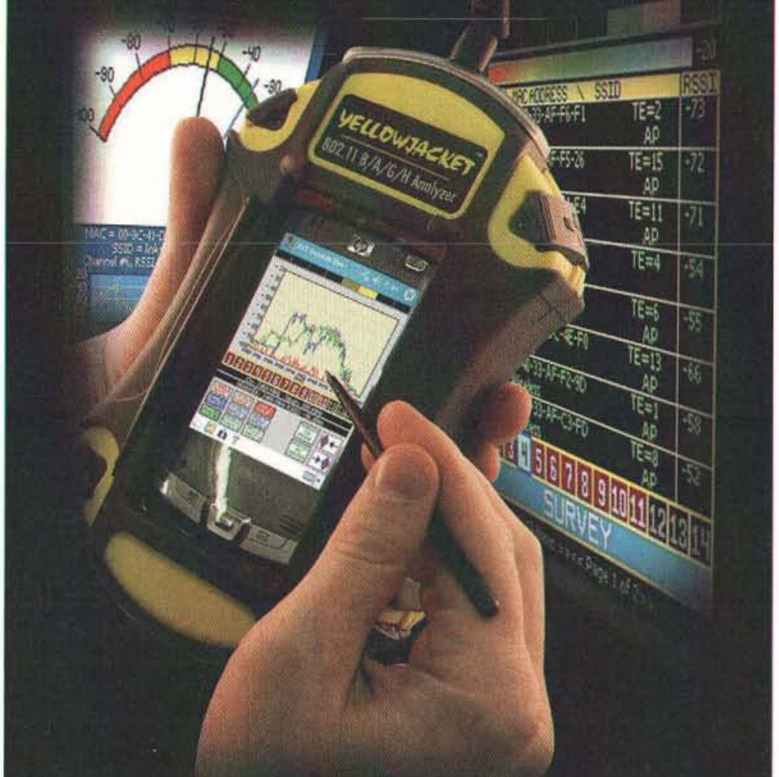
Your Mac Our Patient

HARD DRIVES	BATTERIES	MEMORY	KEYBOARD / KEYS	DRIVE UPGRADES
				
ADD MORE SPACE FOR MUSIC, MOVIES, & ALL OF YOUR FILES.	LONGER LIFE. CHEAP PRICE.	SPEED UP THAT SLOW MAC.	REPLACE THOSE MISSING KEYS.	BURN CDS AND DVDS! UPGRADE YOUR OPTICAL DRIVE.
HARD DRIVES FROM $2.99	BATTERIES FROM $99.95	MEMORY FROM $19.95	REPLACEMENTS FROM $8.95	UPGRADE FOR $119.95

PARTS, UPGRADES, AND REPAIRS FOR YOUR MAC.

WWW.POWERBOOKMEDIC.COM

YELLOWJACKET®-BAGH
802.11b/a/g/h Wi-Fi Analyzer



- Measure 2.4, 5 & 5.7 GHz bands
- Demodulate All 802.11b/a/g/h packets
- User Selectable Power Triggers
- Antenna Alignment & Signal-to-Noise Ratio
- Multipath, MAC, SSID & Absolute Channel
- MAC Security Authorization Lists
- Interference Mapping Software (optional)
- RF Direction Finding (optional antenna)

Also check out these Wi-Fi solutions from BVS

Honeycomb™ handheld & PC software provides RF interference mapping of many networks.

BumbleBee™ handheld spectrum analyzer locates RF interference on any 2.4 or 5 GHz network

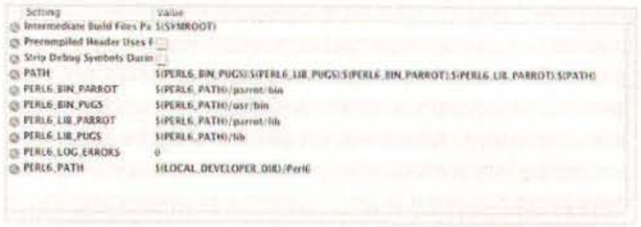
Hive™ indoor mapping software plots AP coverage over any floorplan without GPS.

BERKELEY VARITRONICS SYSTEMS
732-548-3737

Serving the wireless industry for 35 years.

www.bvsystems.com

environmental variable in XCode is then updated to include the locations of the Pugs and Parrot binaries. Finally, the build setting, **PERL6_LOG_ERRORS**, is used to determine whether or not to write the Perl 6 execution errors to a log file. If set to 1, any error messages generated by the Perl 6 script will be written to the log file, `perl6_run.err`.



Setting	Value
Intermediate Build Files Pa	$(SYMROOT)
Precompiled Header Uses F	☐
Strip Debug Symbols Durin	☐
PATH	$(PERL6_BIN_PUGS):$(PERL6_LIB_PUGS):$(PERL6_BIN_PARROT):$(PERL6_LIB_PARROT):$(PATH)
PERL6_BIN_PARROT	$(PERL6_PATH)/parrot/bin
PERL6_BIN_PUGS	$(PERL6_PATH)/usr/bin
PERL6_LIB_PARROT	$(PERL6_PATH)/parrot/lib
PERL6_LIB_PUGS	$(PERL6_PATH)/lib
PERL6_LOG_ERRORS	0
PERL6_PATH	$(LOCAL_DEVELOPER_DIR)/Perl6

Figure 5. Settings of the Perl 6 project template.

The contents of the Run Script phase, Run Perl6, are shown in Listing 3. The script first checks to see if Parrot, Pugs, and the Perl 6 libraries are correctly installed in the OS X system. It then checks the **PERL6_LOG_ERRORS** to see if an error log should be prepared before executing the Perl 6 script. Finally, it starts executing the Perl 6 script by passing the file, **main.p6**, to the **pugs** tool.

Listing 3. The Run Script phase, Run Perl6.

```
# Check to see if Perl 6 is installed
#
# checking for Parrot
if [ ! -x ${PERL6_BIN_PARROT}/parrot ];
then
    echo
    echo "Parrot is not installed on your system."
    exit 1
elif [ ! -x ${PERL6_BIN_PUGS}/pugs ];
then
    echo
    echo "Pugs is not installed on your system."
    exit 1
elif [ ! -d ${LIBRARY_SEARCH_PATHS} ];
then
    echo
    echo "The Perl6 library is not installed on your
system."
    exit 1
else
    # execute the Perl 6 script
    #
    if [ ${PERL6_LOG_ERRORS} -eq 1 ];
    then
        # Initialise the error log
        #
        echo "Log started on:" 1>>perl6_run.err
        date 1>>perl6_run.err

        # Run the Perl 6 script
        #
        pugs main.p6 1>>perl6_run.log 2>>perl6_run.err

        # Finalise the error log
        #
        echo "Log ended on:" 1>>perl6_run.err
        date 1>>perl6_run.err

        # Check the log for errors
        if [ -s perl6_run.err ];
        then
            # check the contents of the Log
            #
            ERR_CNT=`grep -e "[eE]rror" perl6_run.err | wc -
l`
```

WE OUTPERFORM OUR COMPETITION

Nobody beats us on...

» **Price**
Save thousands on multi-location T1 services

» **Performance**
6 to 30× faster than T1 service

» **Reliability**
Best-in-class fixed wireless solutions

Just one of many wireless solutions from Trango...

ATLAS 5010 Series™
Wireless Ethernet Bridge

- » 45 Mbps of sustained throughput
- » Capable of 5.3 and 5.8 GHz
- » Point-to-point, OFDM
- » Up to 40-mile range
- » Adaptable Rate Modulation

Wireless Connectivity IS Powered by Trango

Trango Broadband Wireless
A division of Trango Systems, Inc.

www.trangobroadband.com
Phone: +1 (858) 653-3900

trangobroadband® WIRELESS

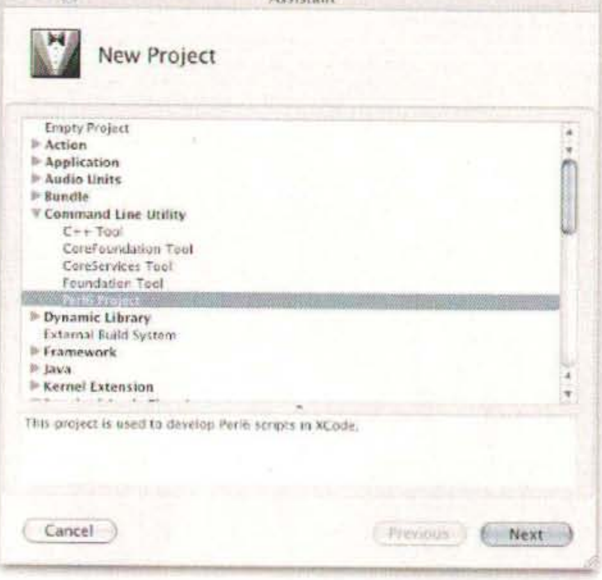
```
            if [ $ERR_CNT -ne 0 ];
            then
                echo "Errors were generated while running the
Perl 6 script."
                exit 1
            fi
        fi
    else
        # Just run the Perl 6 script
        #
        pugs main.p6
    fi
fi
# Successful execution
echo "The Perl 6 script ran without any problems."
exit 0
```

To finalize the project template, the **`.xcodeproj`** bundle is opened using the Finder and two user-specific files containing the file extensions,**`.model`** and **`.pbxuser`**, are deleted. Then a **`TemplateInfo.plist`** file (Listing 4) is created and added to the bundle. This file is used by XCode to identify and preprocess the project template each time it is selected from the New Project Assistant (Figure 6). Once the template is finished, it is then copied to the appropriate subdirectory within

```
    /Library/Application Support/Apple/Developer
Tools/Project Templates/
```

Listing 4. The TemplateInfo.plist file for the Perl 6 project template.

```
{
    FilesToMacroExpand = (
        "main.p6",
    );
    Description = "This project is used to develop Perl6
scripts in ¬
                                XCode.";
}
```


Assistant

New Project

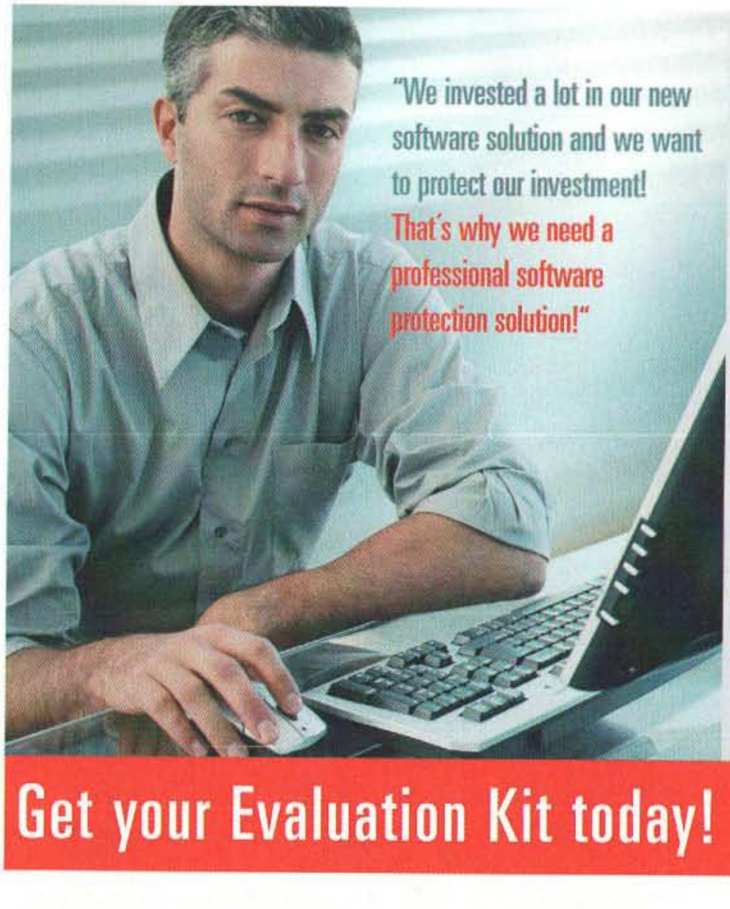
Empty Project
Action
Application
Audio Units
Bundle
Command Line Utility
C++ Tool
CoreFoundation Tool
CoreServices Tool
Foundation Tool
Perl6 Project
Dynamic Library
External Build System
Framework
Java
Kernel Extension

This project is used to develop Perl6 scripts in XCode.

Figure 6. The project template as seen from New Project Assistant.

Concluding Remarks

Perl 6 is the next major revision of the Perl language. It provides a wide variety of new features, as well as long-awaited improvements to the language. It is also the first revision to be designed around an official language specification.



"We invested a lot in our new software solution and we want to protect our investment! That's why we need a professional software protection solution!"

Start protecting your revenue with the most versatile system available!

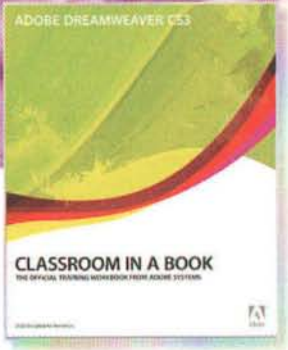
The CRYPTO-BOX ensures that your software is protected reliably against software piracy. The 128-bit AES encryption and the secure internal memory (up to 64 kB) allow you to implement a strong protection mechanism. Multi-platform support allows you to use the same API functions under Windows, Mac OS or Linux.

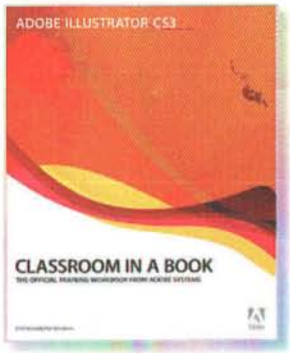
The CRYPTO-BOX Protection Kit includes many features to support your individual protection strategy: protect multiple applications with one CRYPTO-BOX, locally or in networks, and update your licenses remotely via the Internet.

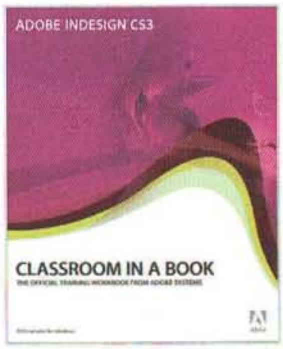
Mac Universal Applications supported

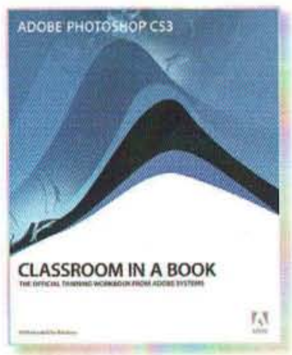
Get your Evaluation Kit today! www.CryptoTech.com/mt or call +1 770 904 0369

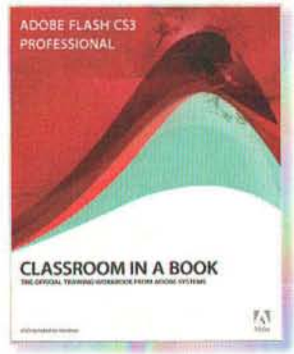
MARX® CryptoTech® LP


ADOBE DREAMWEAVER CS3
CLASSROOM IN A BOOK


ADOBE ILLUSTRATOR CS3
CLASSROOM IN A BOOK


ADOBE INDESIGN CS3
CLASSROOM IN A BOOK


ADOBE PHOTOSHOP CS3
CLASSROOM IN A BOOK


ADOBE FLASH CS3 PROFESSIONAL
CLASSROOM IN A BOOK

Bringing Perl 6 to OS X is accomplished by using XCode to port the Parrot and Pugs projects to the platform. These two open-source projects form the basis of the entire Perl 6 system. Adding Perl 6 support to XCode is also easily accomplished by creating and adding a project template to the development environment. This template contains the basic placeholder file, build settings, and library references that are necessary for developing and running Perl 6 scripts.

Bibliography and References

Randall, Allison, Dan Sugalki, and Leopold Totsh. *Perl 6 Essentials*. Sebastopol, California. (c) 2003. O'Reilly Inc.

Wall, Larry, Tom Christiansen, and Jon Orwant. *Programming in Perl, Third Edition*. Cambridge, Massachusetts. (c) 2000. O'Reilly Inc.

Wikipedia. *Perl*. In Wikipedia, the free encyclopedia. The Wikipedia Community, 2006 May 10. Online: http://en.wikipedia.org/wiki/Perl.

Wikipedia. *Perl 6*. In Wikipedia, the free encyclopedia. The Wikipedia Community, 2006 Apr 26. Online: http://en.wikipedia.org/wiki/Perl_6.

MT

About The Author

JC is a freelance engineering consultant and writer currently residing in North Vancouver, British Columbia. He divides his time between writing technical articles, and teaching origami at his local district's public library. He can be reached at <anarakisware@cashette.com>.

When your customers need a helping hand, don't you want to be there?

With Mac HelpMate Pro, you're just a click away.
www.MacHelpMate.com/testdrive

Mac HelpMate Professional features:

- *Zero-configuration screen sharing (no IP address, no router tweaking necessary). It works through 90% of firewalls and NATs. Reach anyone in the world wherever they are, wherever you are …*
- *Your company's image and logo on the splash screen of the application reinforces your brand, services and sales offerings as well as generates new opportunities*
- *Automated maintenance tasks create billable subscription-based services*
- *One unlimited license covers installation for all of your customers' workstations*
- *Cross platform: Mac - Mac, Mac - Win, Win - Mac and Win - Win*

Mac HelpMate Professional ™ is a product of MOST Training & Consulting
www.macworkshops.com

BMS

THE LAW OFFICE OF
BRADLEY M. SNIDERMAN

Helping clients with their software legal issues.

- **Trademark and Copyright Registration**
- **Trade Secret Protection**
- **Licensing and Non Disclosure Agreements**
- **Assist with Software Audits**

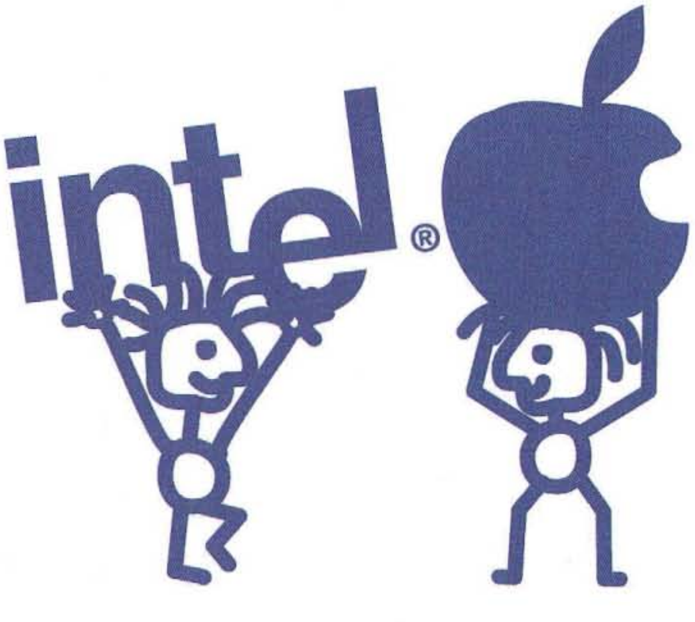
I am an attorney practicing in Intellectual Property, Business Entity Formations, Corporate, Commercial and E-commerce Law.

Please give me a call or an e-mail. Reasonable fees.

23679 Calabasas Rd. #558 • Calabasas, CA 91302
PHONE 818-706-0631 FAX 818-706-0651 EMAIL brad@sniderman.com

Moving to Intel®?

We can deliver.



Mac | Universal

Our technical expertise spans the entire spectrum of software development, including Mac and Windows applications, system-level programming, cross-platform design, handheld and wireless platforms, gaming, web development, and everything in between. We offer comprehensive development services including project management, software design, implementation, and quality assurance. Our work includes commercial projects for worldwide external release, applications for vertical markets, and custom intra-company enterprise software applications. Check us out at www.criticalpath.com | info@criticalpath.com

Critical Path Software

Apple's Transition from Open Firmware to Extensible Firmware Interface

By Criss Myers

Preface

In January 2006, Apple introduced Intel Architecture based Macs. They took this hardware move as an opportunity to also upgrade their choice of firmware. Their previous firmware, Open Firmware, was not in itself a bad choice, being far more advanced than the IBM PC BIOS that Windows computers use. There is, however, a better firmware choice that offers improvements on Open Firmware, called Extensible Firmware Interface, EFI, developed by Intel. When Apple looked for a successor to their PowerPC Processors, they found a complete hardware package from Intel. They got the latest CoreDuo and Xeon Processors, as well as advancements in boot firmware and disk partitioning. Since Apple offers an integrated package, this enables them to control all aspects of the computer system from processors to boot firmware to operating system. This also enabled them to make a smooth transition from the PowerPC, Open Firmware, APM and their 32bit OS, to Intel processors, EFI, GPT and 64bit Tiger OS X. Anyone using these 2 systems will not notice any difference apart from their performance. In this article we will take a brief look at Open Firmware and then explain what EFI is, its history, and what this means for Apple's future as well as that of the PC market in general.

Open Firmware

Apple's PowerPC Macs, post-NuBus, used a boot firmware called Open Firmware. Open Firmware, also called OpenBoot, was developed by Sun Microsystems and is used in Sun's Sparc work stations and servers, IBM POWER systems and PegasosPPC systems and is available under a BSD license. It was described by IEEE as IEEE 1275-1994 but since 1998, it has been withdrawn. In 2006 several commercial versions were released to the open source community under the OpenBIOS project, these include, SUN OpenBOOT, Firmworks OpenFirmware and Codegen SmartFirmware. Open Firmware is a hardware independent firmware and fulfils the same tasks as BIOS does in a PC. The advantage of Open Firmware to Apple is that any I/O cards that work, in say SUN machines, could also be used in Mac machines without requiring any specific Mac drivers. Open Firmware also offered BOOTP capabilities for Netbooting machines as well as setting boot devices. On a Macintosh, most user settings can be changed via GUI tools and the Open Firmware can be locked to prevent booting from non-authorized sources.

Extensible Firmware Interface (EFI)

History

EFI was developed by Andrew fish working for Intel back in the 1990's and was initially called "Intel Boot Initiative (IBI)". It grew up around the need to replace the aging PC BIOS developed by IBM in the 1970's. IBI was intended for their Itanium Architecture (IA) based computers because the existing PC BIOS was far too limiting for future 64 bit operating systems, some of these problems being 16 bit processor mode and PC AT hardware dependencies to name a few. Since then, IBI was developed by Intel into EFI and subsequently released by them to the open source community in 2005. It is now called UEFI (Unified Extensible Firmware Interface) and is currently at version 2.1.

Due to the failed performance of these IA machines, EFI never took off as a common replacement for the PC BIOS. In 2006, Apple moved to EFI and was then the only vendor to take advantage of this firmware interface. Linux has supported EFI since 2000 using "elilo" as a boot loader and both Windows and HP-UX support EFI, but all of these only support IA-32 and IA-64 platforms. Windows Vista is expected to support EFI but only in a later service pack and Microsoft's stance is that until EFI becomes mainstream they will not support it.

What is EFI?

EFI was designed to abstract the firmware and hardware layers from the operating system layer. This is so that operating system vendors and developers no longer need to battle with cumbersome 16bit BIOS and constant hardware configurations.

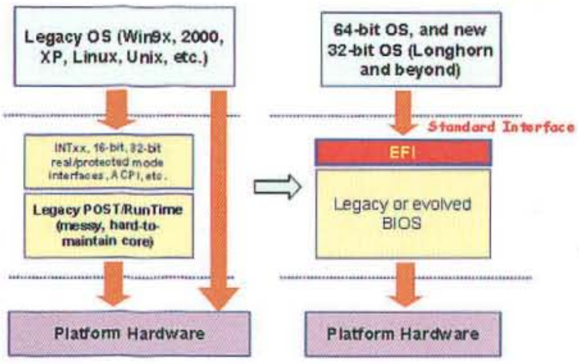

Legacy OS (Win9x, 2000, XP, Linux, Unix, etc.)
64-bit OS, and new 32-bit OS (Longhorn and beyond)
Standard Interface
INTxx, 16-bit, 32-bit real/protected mode interfaces, ACPI, etc.
Legacy POST/RunTime (messy, hard-to-maintain core)
EFI
Legacy or evolved BIOS
Platform Hardware
Platform Hardware

Fig 1. Legacy PC BIOS vs. EFI

Figure 1 compares EFI and BIOS showing EFI as a standard interface between the OS and the hardware. EFI is a set of specifications to define interactions and programmatic interfaces between the hardware and the operating systems, it is then up to the specific hardware vendors to make an EFI compliant system. Since EFI is just a set of specifications, it is not an actual firmware; therefore, it is up to each vendor to create their own firmware to initialize their hardware. Intel's firmware, and hence Apple's choice, is called "Intel Platform Innovation Framework for EFI", also called "Framework". It is a legacy free firmware that complies with EFI. It is not available to end-users as a complete firmware package but parts of the code are released under the TianoCore project, (www.tianocore.org). Apple created their own version of "Framework"; we will see later how they implemented EFI and framework.

EFI is very similar to Open Firmware in that it is a boot firmware independent of hardware and operating system. What this means is that it allows the operating system to boot and run in a "sandbox" mode. A "sandbox" mode is a safe protected mode where the operating system does not make direct calls to the hardware. EFI controls the hardware; it takes the calls from the operating system, and then passes them on to the hardware, creating an interface between the OS and the physical firmware, in much the same way that a virtual machine works. This then offers a stable environment from which to run the OS. The advantage of this is that only the firmware controls the hardware (see Fig 2).

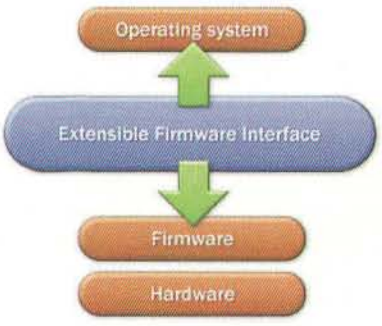


Figure 2 - EFI as an interface between the OS and the Firmware.

For developers this means they will no longer need to make BIOS/OS dependant drivers, they can create EFI based drivers which will then work in any compatible EFI machine irrelevant of the OS or BIOS loaded. Obviously, for Apple this means a wider range of I/O cards can be supported in the future once more hardware vendors support EFI. EFI is free of any of the memory restrictions that BIOS has and can use all available memory. This speeds up boot time as well, a Mac Pro for instance can boot up and load the OS in just 15 seconds.

The EFI specification is broken down into "boot services" and "runtime services". Boot Services are any services that run during boot only, such as, the loading of drivers and the accessing of graphics during boot. The services use "EFI Drivers" "EFI Applications" and "EFI Boot Code". Runtime Services run while the computer is running accessing such things as date/time etc. EFI Drivers are written in C and conform to the EFI Drivers Model. They can be loaded from any non-volatile memory, either in option ROM, or on the device directly.

Apple's implementation of EFI and Framework

The EFI Boot Process

Firstly, the computer powers up, the Framework then, via EFI boot services, initializes the hardware such as the Bluetooth, USB, VGA, network IP stack, remote control, etc. These boot services will load all the hardware drivers necessary to detect any OS that resides on either an internal or external hard disk, or a network volume. Apple's Framework does not boot into text mode but directly into graphics mode, just like it did with Open Firmware. There is no direct access to the text mode or EFI shell. Apple has developed their Framework to work in much the same way as Open Firmware did**,** using the same key strokes: C boots to CD/DVD, N boots into the Netboot disk, V boots verbose and S boots into Safe Mode, etc. This makes a seamless transition from Open Firmware to EFI for the user.

Following this, various EFI Applications can be loaded; one such application is the Boot Manager. The Boot Manager is used to select and load the operating system, removing the need for a dedicated boot-loader mechanism. On an Apple this looks very similar to the Open Firmware boot-loader but with updated graphics. Apple has created their own EFI compliant Boot Manager with their own graphics. Apple's Boot Manager can detect any Mac operating system that is available to the system. According to Apple, their EFI Application can, however, only read boot-code from a GPT formatted drive or an Intel based NetBoot image; only one Netboot image is displayed and is either set via the "Startup Disk" section on the client or is the default image set on the server. Initially the Boot Manager could only detect Mac OS, but Apple used the "compatibility support module" to support legacy PC BIOS such as XP. This will only display a single legacy OS per drive.

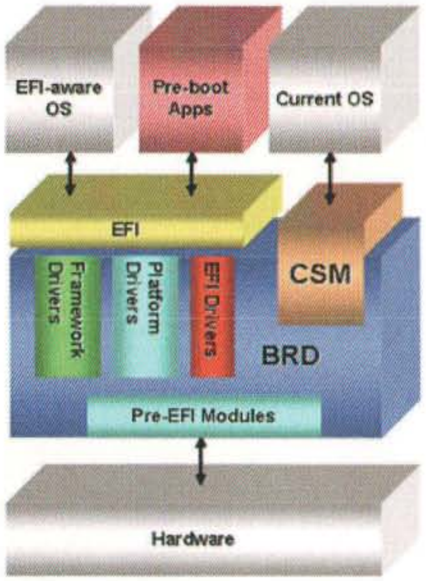
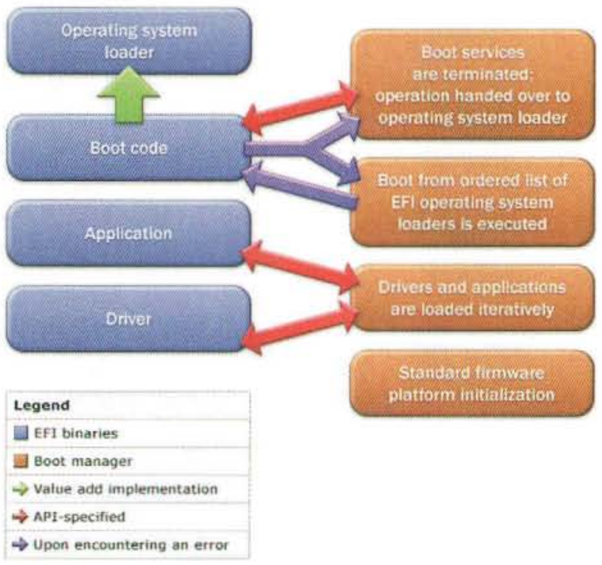


Figure 3 - EFI with the CSM Module.

Apple's Boot Manager Application will either boot from the default volume or you can enter the graphical selection screen via the option keystroke, just as with Open Firmware. The EFI boot code is then read from the hard disk and the operating system is loaded. Control is then transferred to the OS. Some higher operating system drivers can now be loaded for various devices. The operating system will then pass calls to the firmware, which will pass them to the hardware.



Operating system loader
Boot code
Application
Driver
Boot services are terminated; operation handed over to operating system loader
Boot from ordered list of EFI operating system loaders is executed
Drivers and applications are loaded iteratively
Standard firmware platform initialization
Legend
EFI binaries
Boot manager
Value add implementation
API-specified
Upon encountering an error

Figure 4 - EFI Boot Process.

rEFIt tools

When Apple released BootCamp with their Firmware update, which added the compatibility support module to their firmware, it became possible to load the Linux OS onto a Mac. However, this module only recognizes a single legacy OS per drive and labels them as Windows only. This is not useful in a triple boot system. An open source project was then started called rEFIt tools which offers a boot menu to detect multiple OS per drive and label them as the appropriate OS's. It also offers a maintenance toolkit with direct access to the EFI pre-boot environment and gives access to the EFI Shell Application. The current version is 0.8 and can be loaded via CD, USB or directly off the internal drive. When you boot the Mac with the option key held down, you can then select the rEFIt tools from their installed location.

Conclusion – Future Benefits of EFI for Apple

EFI is a complete pre-boot environment, that makes life much easier for developers, and since EFI is written in C, it is much easier to program for. With more vendors supporting this specification it means, for Apple users, that more and more devices with be compatible with Macs. This can only be a good thing. Developers find it a much easier environment to develop in due to its pre-boot environment. They can test and develop without the need for a VDU because they can output the tests directly to text. The pre-boot environment can also be used for backup, recovery, and diagnostics, updating firmware by accessing the Internet without the need for an OS. Both American Megatrends Inc (AMI) and Insyde Technology, both members of UEFI, have made Pre-Boot Applications sets.

Figure 5 - AMI Pre-Boot Interface.

There are development tools available that allow users to create their own pre-boot applications.

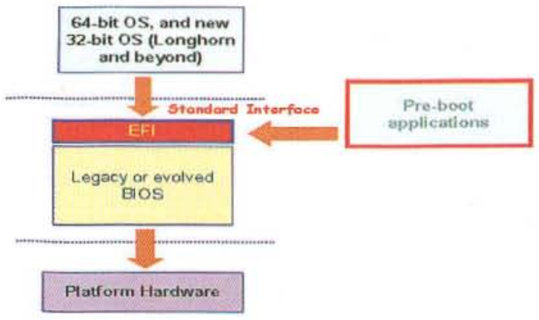


Figure 6 - Pre-Boot Applications.

NetTeam SERVER

Susbsription service now available, running on Mac OS X Server.
Details at: **www.netteamserver.com/subscribe**

Connect your people, contacts, relationships, projects, tasks, documents, blogs, web content ... your knowledge itself ... and magic follows.

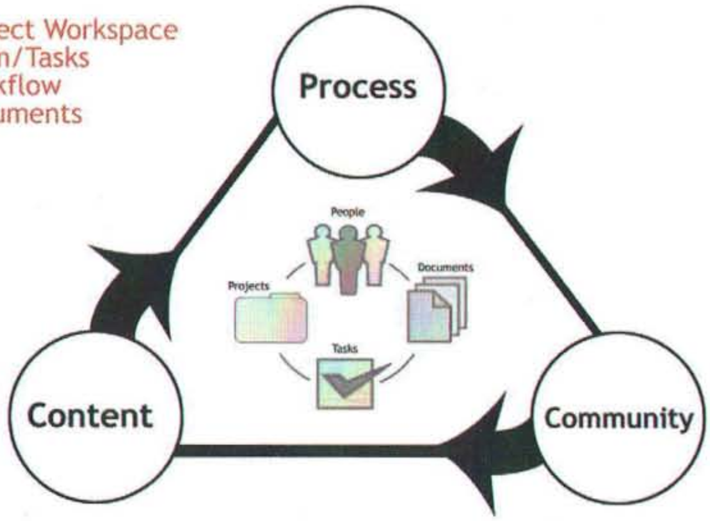
Most businesses already have this data but in different systems, used by different people, in different departments. What a nuisance! **NetTeam Server brings it all together, in a web app that everyone can use, anywhere.** What a difference!

NetTeam Server

NetTeam Server is a business process, content management, collaboration and social networking web app for businesses and organizations of any size. It offers people, project, task and document management services and has a powerful API to support customization.

The triangle represents NetTeam Server's unique combination of functionality for three critical areas: **Process, Content** and **Community.** We're a good choice even if you only need one of these, but if your business operations embrace two or three, you'll love what we can do for you.

- Project Workspace
- Team/Tasks
- Workflow
- Documents



- Web Content/Site Management
- Web Document Library
- Project and Shared Blogs
- News Editor

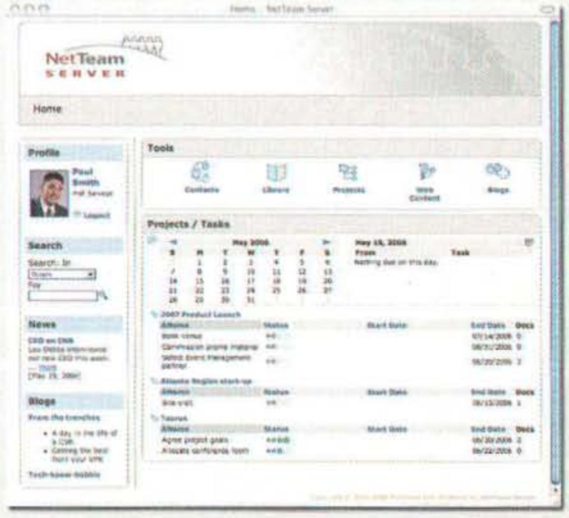
- Users, roles and relationships
- Simple CRM
- Social Networking
- Profile & Personal Blogs

Features

NetTeam Server's fundamental constructs are **People, Projects, Tasks** and **Documents.**

These four are central to all business activities, so we bring them together in a coherent workspace that makes NetTeam Server a true *Business Operating System*. The portal interface can be tailored to match client branding and linked systems, and includes five Editors (see screenshot) and a modern, AJAX-enhanced, configurable user interface.

User roles determine access privileges and which (if any) tools are presented on login. Blogs are used extensively to support publishing, information and knowledge management. Wikis will be available in a late-summer update.



NetTeam Server is available for Mac OS X Server, Linux and Windows platforms and supports all leading web browsers. A Web Services API allows tight integration with other systems and single sign-on. We also offer a Java mobile client which can be tailored to support mobile workforce applications.

NetTeam Server has been successful in deployments serving from 10 to 10,000 users and may be installed on a server of your choosing, used on a dedicated server we provide, or rented as a subscription service (multi-company server).

Consultant, reseller and developer enquiries welcome.

NetTeam • www.netteamserver.com
2300 M St NW, Suite 800, Washington DC 20037, USA
+ 1 (202) 250 3434 • info@netteamconsulting.com

This offers Apple the chance to create their own Pre-Boot Applications which can be stored on the System EFI partition, which is currently empty on Apples GPT drives, but is created by default with a set size of 200mb. Apple could develop pre-boot versions of Time Machine for Leopard so that an OS X system could be recovered via an existing backup, or use a pre-boot Disk Utility to repair a drive. Third party companies such as Norton or Tech Tool could install pre-boot versions of their Applications to this hidden partition also. Apple could also make a pre-boot version of Front Row so that users can access their DVD's, MP3s, AppleTV without needing to boot the OS. This also offers secure network booting, remote provisioning and setup as well as virus scanning etc. For the general user it will offer them a stable environment to run the Mac OS X without as many kernel panics.

Bibliography and References

UEFI. UEFI Extensible Firmware Interface Specifications Version 2.1, Copyright 2006.

Windows Hardware Developer Central, EFI and Windows, April 2006.

Wikipedia, Extensible Firmware Interface, 2007.

Wikipedia, Open Firmware, 2007.

Wikipedia, BIOS, 2007.

OSX86Project, EFI, February 2007.

Intel Software Network, Extensible Firmware Interface, 2006.

Intel Software Network, Enhanced Pre-Boot Environments with EFI Applications, 2007

Michael Kinney - Intel Developer Update Magazine, Solving Boot Issues with EFI, September 2000

Amit Singh - Kernalthread.com, More Power to Firmware, Copyright 2006.

American Megatrends Inc, AMI Pre-Boot Applications, December 2005.

MT

About The Author

Criss Myers is a Senior Mac IT Technician for the Faculty of Science and Technology, at the University of Central Lancashire, Preston, United Kingdom. He has been a Systems Server Administrator from the very first version of OS X Server. He Works with Macs as well as Linux, Unix and Windows.

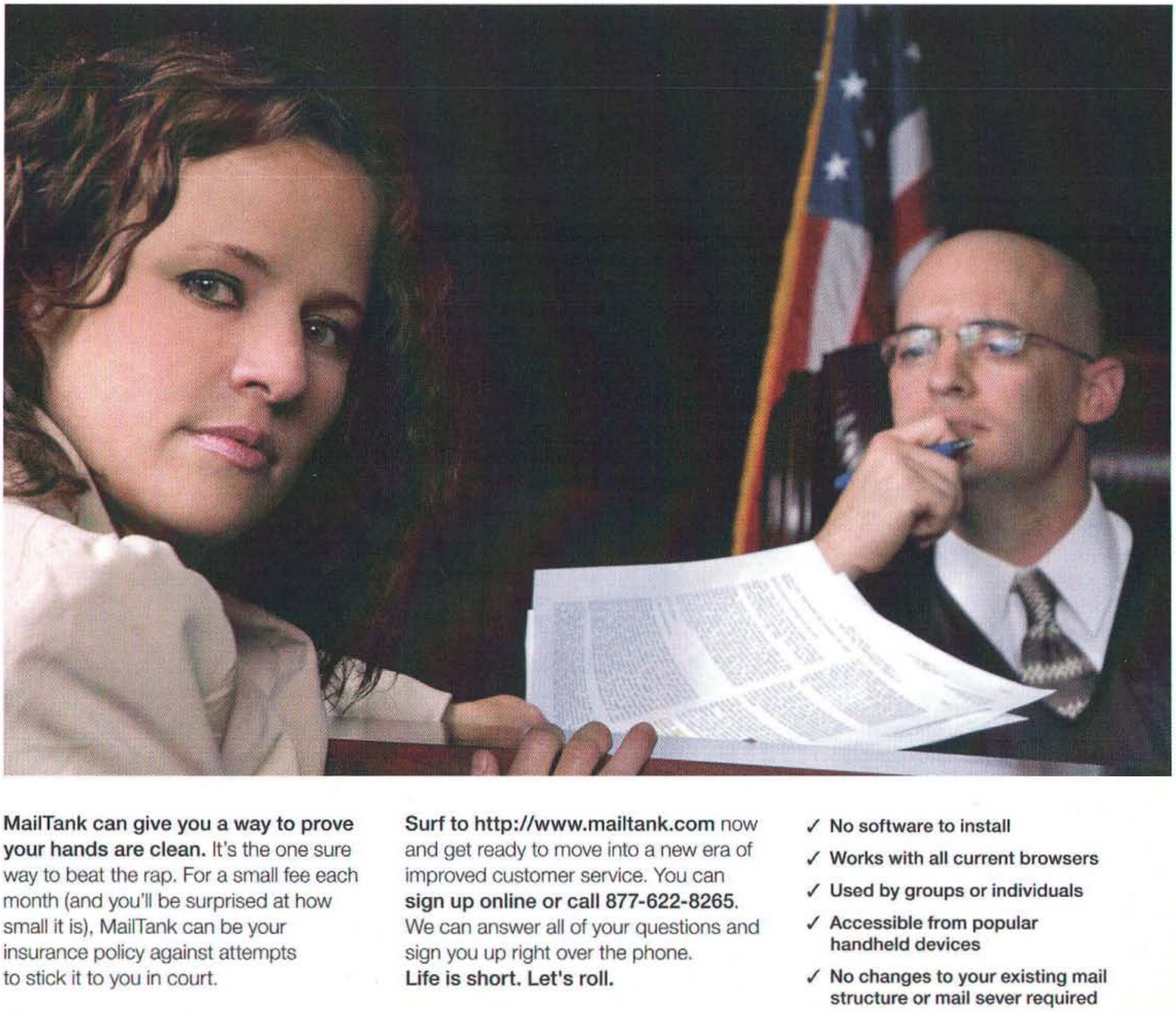
Are you ready for ediscovery? MailTank can help you beat the rap!

eDiscovery is here to stay. Managing business communications is now a lot more difficult. If you've been following the news, you know that missing email is a red flag for zealous investigators wanting to make a case against you.

No one who's running a business since the new regulations kicked in can be sure when cyber-sleuths from the government -- or a private party nursing a grievance – may show up at your door with a subpoena in hand.

All of us have learned by now that we need to be careful with email. Even the most innocent exchanges might be taken out of context and may be used against you. If an eager prosecutor or attorney decides to launch a fishing expedition, don't be surprised – be prepared.

If one day you are asked to produce all of the email you've ever exchanged with a customer, prospect, lawyer, employee -- or just about anyone you've done business with, will you be ready?



MailTank can give you a way to prove your hands are clean. It's the one sure way to beat the rap. For a small fee each month (and you'll be surprised at how small it is), MailTank can be your insurance policy against attempts to stick it to you in court.

Surf to http://www.mailtank.com now and get ready to move into a new era of improved customer service. You can **sign up online or call 877-622-8265.** We can answer all of your questions and sign you up right over the phone. **Life is short. Let's roll.**

- ✓ **No software to install**
- ✓ **Works with all current browsers**
- ✓ **Used by groups or individuals**
- ✓ **Accessible from popular handheld devices**
- ✓ **No changes to your existing mail structure or mail sever required**

MailTank

www.mailtank.com - 877-622-8265 - support@mailtank.com - powered by MacTank

Next month: Work from anywhere - Desktop to iPhone

Uninstalling with AppleScript

Building an uninstaller with AppleScript Studio

By José R.C. Cruz

Introduction

In a previous MacTech article, we learned how to use the PackageMaker tool to build a distribution package. We also learned how to localize that package for different languages, and how to customize it with scripts. But the one thing we were unable to do with the tool is to build an uninstaller.

Third-party tools such as ViseX and InstallerMaker can add an uninstall option to their installers. This feature is sadly missing from PackageMaker, though not without reason. In fact, most products are easy to remove – they are contained in a single folder, and trashing that folder completely uninstalls the product. But some products will have multiple files installed in different directories. Others will create custom directories to store the files. A good example of such a product is **Xcode**. Its installer creates the custom directory `Developer` to contain its files. The installer also stores files in other directories such as `/System/Library` and `/Library/Application Support`. As a result, removing Xcode from the system can be a laborious process. In these situations, an uninstaller tool can be useful.

This article will demonstrate how to use AppleScript Studio to build an uninstaller. To get readers started on their own uninstallers, the Xcode project **Uninstall** is made available for downloading. A copy of this project can be obtained at the following URL: <ftp.mactech.com/src/mactech/volume23_2007/23.05.sit>.

The Receipts Bundle

After the Installer installs a software payload, it creates a copy of the package in the Receipts directory. The path to this directory is `/Library/Receipts` on the boot volume.

The copy of the installer package is known as a *receipt bundle*. Its presence indicates that a product has been installed successfully. It also tells the installer package if the latter has to do an upgrade, as opposed to an installation.

Figure 1 shows the internal structure of a typical receipt bundle. Notice that the bundle has most of the same files as an installer package. What is missing is the `.paz.gz` file containing the actual payload. Also missing are the two aliases to that file from the `Resources` subdirectory.

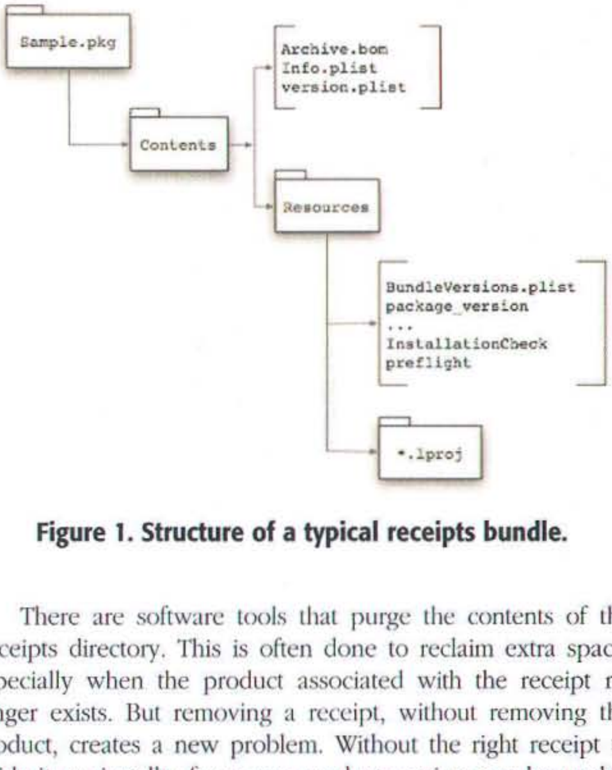


Figure 1. Structure of a typical receipts bundle.

There are software tools that purge the contents of the Receipts directory. This is often done to reclaim extra space, especially when the product associated with the receipt no longer exists. But removing a receipt, without removing the product, creates a new problem. Without the right receipt to guide it, an installer for a new product version may be unable to upgrade the current product correctly.

The BOM File

The BOM file is a list of all the files that comprise the software payload. It also defines the locations of each file on the target volume. This file is present in both the installer package and receipts bundle.

The BOM file format has its origins in the NeXTStep operating system. It is also a binary format and, as a result, is not directly readable. To read its contents, use the command-line tool **lsbom** to preprocess the BOM file.

The lsbom tool

The **lsbom** tool takes a BOM file as its input and renders its contents into human readable text. It then outputs the text to another file, or to **stdout** by default. The tool is a standard addition to the BSD subsystem of MacOS X. Its counterpart is the **mkbom** tool, which creates a BOM file for a given directory.

To use the **lsbom** tool, simply pass the path to the BOM file as its input. For example, to process the BOM file for the Sample.pkg receipt, type the following statement at the Terminal prompt.

```
lsbom /Library/Receipts/Sample.pkg/Contents/Archive.bom
```

The tool will then parse the file and display its contents at lightning speed on the Terminal speed. To better read the output, pipe the results of the **lsbom** tool to the **less** tool.

```
    lsbom /Library/Receipts/Sample.pkg/Contents/Archive.bom |
less
```

less will display the first **N** lines of text from **lsbom**. The number of lines displayed is dictated by the height of the Terminal window. To display the next **N** lines of text, tap on the Space bar. To display the previous **N** lines of text, tap on the **B** key while holding down the **<CTRL>** key.

Another way of handling the **lsbom** output is to save it to a file. To save the output to the file **Sample.log**, use the I/O redirection token **'>'**.

```
    lsbom /Library/Receipts/Sample.pkg/Contents/Archive.bom >
Sample.log
```

Notice that, in both examples, the BOM file is always inside the **Contents** subdirectory of the receipt bundle. Also, the name of the BOM file is always **Archive.bom**. Most receipt bundles will follow the same conventions. Though it is possible for a BOM file to be located elsewhere in the bundle, this is rarely done.

The lsbom output

Listing 1 shows a sample output from the **lsbom** tool. Each entry corresponds to a file or directory installed by the package. The first three items of the BOM entry are arranged as follows.

directory_path file_modes user_id/group_id.

Each item is separated from the other by a single tab character (**0x09**). Now, if the entry is for a *regular file*, it will have two more items as shown below.

```
    directory_path file_modes user_id/group_id
number_of_bytes crc_32
```

If it is for a *symbolic link*, its last item will be the *path to the original file or directory*.

```
    directory_path file_modes user_id/group_id
number_of_bytes ¬
            original_path
```

Finally, if the entry is for a *device file*, its last item will be the assigned *device number*.

```
    directory_path file_modes user_id/group_id device_number
```

Listing 1. Sample output of the lsbom tool

```
.  40755  501/80
./Sample.app   40755   501/80
./Sample.app/Contents  40755   501/80
./Sample.app/Contents/Info.plist  100644  501/80  947 ¬
      1163649540
./Sample.app/Contents/MacOS   40755   501/80
./Sample.app/Contents/MacOS/Sample    100755  501/80  51956 ¬
      3193247409
./Sample.app/Contents/PkgInfo 100644  501/80  8   742937289
./Sample.app/Contents/Resources   40755   501/80
./Sample.app/Contents/Resources/Appearance.tiff  100644
501/80  4850    1276692542
...
./Sample.app/Contents/Resources/English.l
proj  40755 501/80
./Sample.app/Contents/Resources/English.lproj/Credits.rtf
100644  501/80  3163    3602318773
./Sample.app/Contents/Resources/English.lproj/Errors.strings
100644  501/80  2978    2815210102
...
```

Most BOM file listings consist mostly of directories and regular files. Device files and symbolic links are seldom found. Also, the file_modes item is essentially the *three permission flags* written in octal form. Each of the lower three numbers represents the permission for *world, group*, and *owner*. The upper set of numbers represent the type of item in question. They are set to 40 for a directory, 100 for a generic file.

Notice that each **directory_path** item starts with a dot **(.)** character. This character is replaced by the **IFPkgRelocatedPath** value set in the **Info.plist** file. If that value is not set, the directory path is assumed to be relative to the OS X boot volume.

Also, if the package has installed payloads in other directories, the BOM listing will show these payloads. For example, if **Sample.pkg** has installed two files in the **/usr/bin** directory, the BOM listing may show these files as follows.

```
    ./usr/bin           40755       0/0
    ./usr/bin/foo      100755  0/0 12606   2275820725
    ./usr/bin/fubar    100755  0/0 12606   2275820725
```

Notice that both **user_id** and **group_id** items are set to 0 in the above example. This means that the owner of the two files and the directory is **root**. Removing the two files will require authentication. Do not, however, remove any directories or subdirectories with a root owner. Doing so may remove important files, and render the entire OS X platform unusable.

The lsbom options

The **lsbom** tool also provides a number of output options. Use these options to display specific entries from the BOM file as follows.

To display only the directories accessed or created during installation, use the **-d** option.

```
    lsbom -d
/Library/Receipts/Sample.pkg/Contents/Archive.bom
```

The output listing will also include bundles such as **.app**, **.bundle**, and **.lproj**. To display only the paths of files that were installed or updated, use the **-f** option.

```
    lsbom -f
/Library/Receipts/Sample.pkg/Contents/Archive.bom
```

To display only the paths of each directory and file, use the **-s** option.

```
lsbom -s
/Library/Receipts/Sample.pkg/Contents/Archive.bom
```

The `lsbom` tool also has options other than the ones shown above. To view a list of options, type the command `lsbom –h` at the Terminal prompt. Also, to view the tool's electronic manual, type `info lsbom` at the prompt.

AppleScript and the Shell

AppleScript is the native scripting language of the MacOS platform. First introduced in the 1990s, it is one of the first few languages that work in a GUI environment. It also uses a natural language syntax, which makes its scripts easy to read and write.

Another feature of AppleScript is that it can be extended using plug-ins. These plug-ins, or *scripting additions*, allow AppleScript to do tasks that are slow or impossible to do using the core language. The OS X version of AppleScript comes bundled with the plug-in named **Standards Additions**. With this plug-in, an AppleScript script can display simple dialogs and perform basic file I/O tasks. The script can also run Unix shell scripts using the **`do shell script`** function.

The `do shell script` function

The `do shell script` function is AppleScript's gateway to the BSD subsystem of MacOS X. With this function, an AppleScript script can execute command-line tools or shell script files. It can also run a single-line shell script using this function. The function returns any results from the script as a string.

The function uses the interpreter set by the `SHELL` environment variable to do its tasks. To find out the current interpreter, launch the Script Editor tool, which is located in `/Applications/AppleScripts`. On the script window, type `do shell script "printenv SHELL"` and click on the **Run** button. If the current interpreter is `bash`, the function will return the string value of `SHELL=/bin/bash` on the **Results** pane.

Working with file paths

When using the do shell script function to manipulate files, it requires the file paths expressed using the POSIX format. In short, a forward slash character </> must separate each path name. For example, to parse the BOM file for Sample.pkg, pass the script to the function as follows.

```
do shell script ¬
    "lsbom
/Library/Receipts/Sample.pkg/Contents/Archive.bom"
```

Now if a path name contains any spaces, a reverse slash <\> character must precede each space. For example, to parse the BOM file for `Test Sample.pkg`, pass the script as follows.

```
do shell script ¬
    "lsbom /Library/Receipts/Test\\
Sample.pkg/Contents/Archive.bom"
```

Notice that *two* reverse slashes precede the space in `Test Sample.pkg`. This is necessary due to a little quirk in AppleScript. The first reverse slash tells AppleScript to treat the second slash is part of the string. The second slash tells the shell interpreter to treat the space as part of the script text.

AppleScript, however, expresses its file paths using the MacOS format. Instead of a forward slash, each path name is separated by a colon <:> character. Also, each path name can contain spaces without the need for any reverse slashes. For example, the file path to the BOM file for `Test Sample.pkg` is written in MacOS format as follows.

```
OS X:Library:Receipts:Test
Sample.pkg:Contents:Archive.bom
```

Converting between file path formats can be quite tedious. To address this issue, the Standards Additions plug-in provides the POSIX file class. To convert the MacOS file path to `Sample.pkg` to the POSIX format, type the following statement in the Script Editor window.

```
POSIX path of alias "OS X:Library:Receipts:Sample.pkg"
```

This will return the converted path as `/Library/Receipts/Sample.pkg`. To convert it back to a MacOS format, type the following on the editor window.

```
POSIX file "/Library/Receipts/Sample.pkg"
```

Both examples have file paths set relative to the boot volume. Both also assume that the MacOS name of the boot volume is `OS X`. Now if a file path is set relative to a volume other than boot, the conversion will reflect that volume. For example, if the MacOS file path is set to `Users:Applications:Public:`, it will be `/Volumes/Users/Applications/Public/` in POSIX format.

Authenticating a command

Some shell commands require authentication in order to perform their tasks. They are usually invoked in the Terminal window using the `sudo` command. For example, to create the subdirectory `foo` in `/usr`, type the following line at the Terminal prompt.

```
sudo mkdir /usr/foo
```

The `sudo` command first prompts the user for an administrative password. When the correct password is entered, `sudo` then executes the `mkdir` command. Otherwise, it aborts after the user fails to enter the right password thrice in a row.

Using the `sudo` command through the `do shell script` function is both tedious and unnecessary. Instead, the function can authenticate the desired command by itself. For example, to create the same subdirectory shown above, type the following line on the Script Editor window.

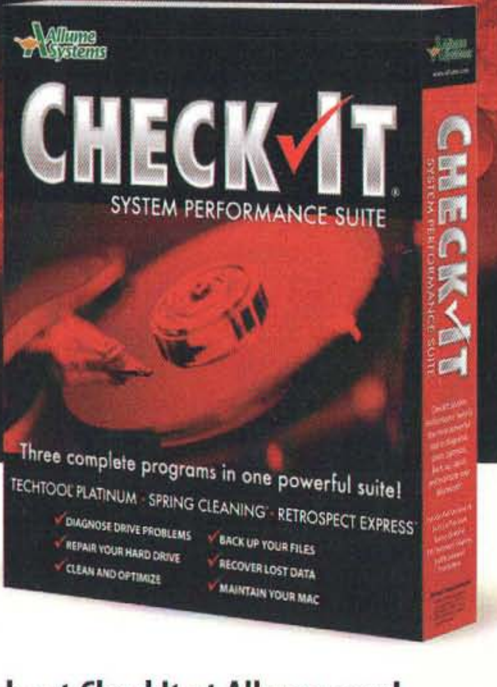
```
do shell script "mkdir /usr/foo" with administrator
privileges
```

The function first prompts the user for a password using the dialog shown in Figure 2. Again, when the user enters the correct username and password, the function then executes the mkdir command. Otherwise, it aborts with an error message after the user fails to enter the right information three times in a row. The same also happens if the user clicks on the **Cancel** button.

DON'T LET YOUR MAC WIND DOWN!

CLEAN UP YOUR MAC
UNINSTALL APPLICATIONS
BOOST PERFORMANCE
REGAIN LOST DISK SPACE

INCLUDES FULL VERSIONS OF:
TECHTOOL PLATINUM
SPRING CLEANING
DANTZ RETROSPECT EXPRESS

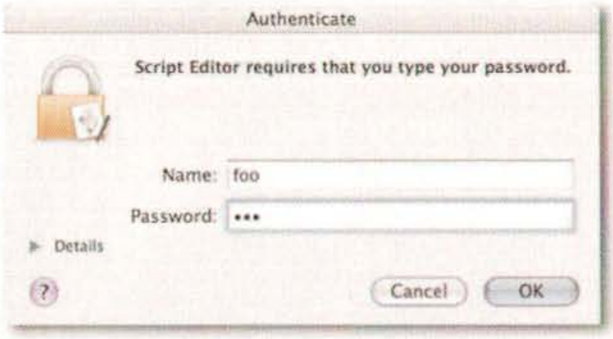


Allume Systems

A Division of Smith Micro Software

Available from your favorite retailers and catalogs. Learn more about CheckIt at Allume.com!

CheckIt System Performance Suite is the most powerful tool to diagnose, clean, optimize, back up, and repair your Macintosh!


Authenticate

Script Editor requires that you type your password.

Name: foo
Password: •••

Figure 2. The authentication dialog.

To use either approach, make sure that you have a user account *with administrative privileges*. To learn how to create such an account, consult one of the references listed at the end of this article.

Building with AppleScript Studio

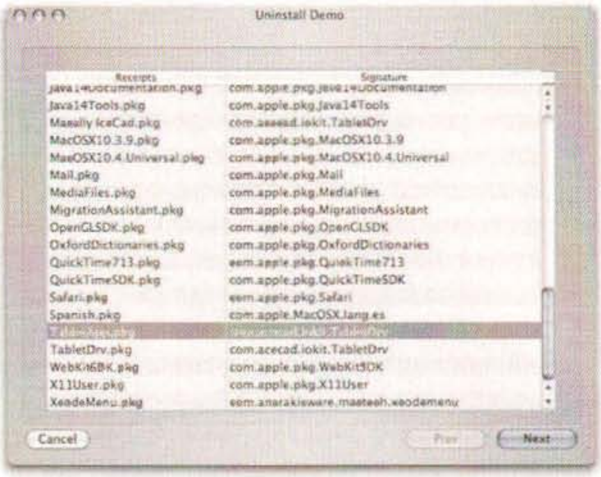
There are many ways to build an uninstaller. One way is to build it as a *shell script*. One example is Xcode, which comes with a Perl script to uninstall its various components. This approach is easy to implement and test. It does, however, require the use of the Terminal window. It also provides very poor user interaction and feedback, if any.

Another way is to build the uninstaller as a *Cocoa application*. Cocoa gives the uninstaller a better way of interacting with the user. It also allows the uninstaller to perform tasks not possible with a shell command. But this approach requires too much resources and time to implement. It also has a very high learning curve.

A more practical way is to build the uninstaller as an AppleScript application. This is now easy to do with AppleScript Studio. The uninstaller gets a decent interface with which to interact with the user. It will be easy to build and test due to AppleScript's user-friendly syntax. This is the approach used for the Xcode project **Uninstall**.

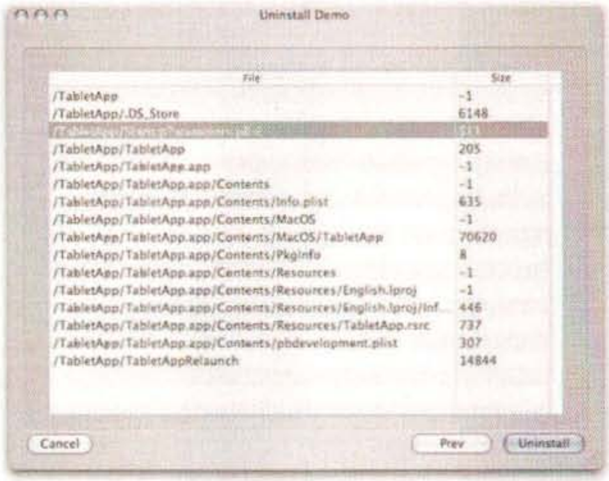
Laying out the user interface

The Uninstall project has a single main window named **Uninstall Demo**. The window is subdivided into two panels by an NSTabView control. Both panels contain a single NSTableView control. The first table displays the contents of the `Receipts` directory (Figure 3). The second table displays the contents of the BOM file for the selected receipt (Figure 4).


Uninstall Demo

Receipts	Signature
Java14Documentation.pkg	com.apple.pkg.Java14Documentation
Java14Tools.pkg	com.apple.pkg.Java14Tools
Maeally IceCad.pkg	com.aeeead.iokit.TabletDrv
MacOSX10.3.9.pkg	com.apple.pkg.MacOSX10.3.9
MacOSX10.4.Universal.pkg	com.apple.pkg.MacOSX10.4.Universal
Mail.pkg	com.apple.pkg.Mail
MediaFiles.pkg	com.apple.pkg.MediaFiles
MigrationAssistant.pkg	com.apple.pkg.MigrationAssistant
OpenGLSDK.pkg	com.apple.pkg.OpenGLSDK
OxfordDictionaries.pkg	com.apple.pkg.OxfordDictionaries
QuickTime713.pkg	eem.apple.pkg.QuiekTime713
QuickTimeSDK.pkg	com.apple.pkg.QuickTimeSDK
Safari.pkg	eem.apple.pkg.Safari
Spanish.pkg	com.apple.MacOSX.lang.es
[illegible]	[illegible]
TabletDrv.pkg	com.acecad.iokit.TabletDrv
WebKit6BK.pkg	com.apple.pkg.WebKitSDK
X11User.pkg	com.apple.pkg.X11User
XeodeMenu.pkg	eem.anarakieware.maeteeh.xeodemenu

Figure 3. The Uninstall window, Receipts panel.


Uninstall Demo

File	Size
/TabletApp	-1
/TabletApp/.DS_Store	6148
[illegible]	[illegible]
/TabletApp/TabletApp	205
/TabletApp/TabletApp.app	-1
/TabletApp/TabletApp.app/Contents	-1
/TabletApp/TabletApp.app/Contents/Info.plist	635
/TabletApp/TabletApp.app/Contents/MacOS	-1
/TabletApp/TabletApp.app/Contents/MacOS/TabletApp	70620
/TabletApp/TabletApp.app/Contents/PkgInfo	8
/TabletApp/TabletApp.app/Contents/Resources	-1
/TabletApp/TabletApp.app/Contents/Resources/English.lproj	-1
/TabletApp/TabletApp.app/Contents/Resources/English.lproj/Inf...	446
/TabletApp/TabletApp.app/Contents/Resources/TabletApp.rsrc	737
/TabletApp/TabletApp.app/Contents/pbdevelopment.plist	307
/TabletApp/TabletAppRelaunch	14844

Figure 4. The Uninstall window, Files panel.

The entire window layout is that of a basic Assistant. On the lower right corner are two pushbuttons, **Prev** and **Next**. The **Next** button is also set as the default button.

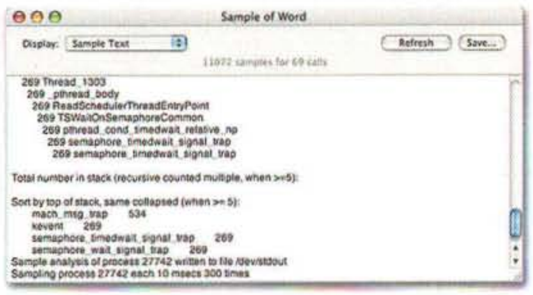
Both buttons are disabled by default. The **Next** button is enabled when the Receipts table has a selected entry. The **Prev** button is enabled when the current panel is the one with the Files table. Also, when the Files table has a selected entry, the **Next** button becomes the **Uninstall** button.

On the lower left corner of the window is the **Cancel** button. This button is set to respond to the `<Esc>` key. It also sends a `performClose:` message to the window when clicked.

Binding the widgets

The window and some of the controls are then bound to specific AppleScript handlers. The bindings are set in the AppleScript panel of the **Show Inspector** dialog (Figure 5). To display the dialog, choose **Show Inspector** from the **Tools** menu. Then select **AppleScript** from the drop-down list at the top of the dialog.

Does



```
269 Thread_1303
 269 _pthread_body
  269 ReadSchedulerThreadEntryPoint
   269 TSWaitOnSemaphoreCommon
    269 pthread_cond_timedwait_relative_np
     269 semaphore_timedwait_signal_trap
      269 semaphore_timedwait_signal_trap

Total number in stack (recursive counted multiple, when >=5):

Sort by top of stack, same collapsed (when >= 5):
    mach_msg_trap      534
    kevent     269
    semaphore_timedwait_signal_trap      269
    semaphore_wait_signal_trap      269
Sample analysis of process 27742 written to file /dev/stdout
Sampling process 27742 each 10 msecs 300 times
```

+

= questions?

Are you routinely looking for answers?

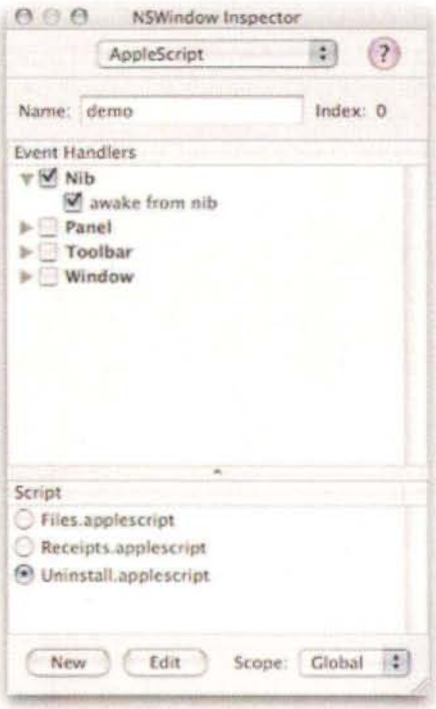
Imagine a whole year of answers.

MacTech Magazine is already read every month by tens of thousands of readers. Readers that represent the very heart and soul of the Mac community. Join the crowd and sign up today!

For a special one year subscription, visit:
store.mactech.com

MACTECH®

Toll Free 877-MACTECH, Outside US/Canada: 805-494-9797


NSWindow Inspector

AppleScript

Name: demo Index: 0

Event Handlers

- [✓] Nib
  - [✓] awake from nib
- [ ] Panel
- [ ] Toolbar
- [ ] Window

Script

- ○ Files.applescript
- ○ Receipts.applescript
- ● Uninstall.applescript

Figure 5. The Show Inspector dialog, AppleScript panel.

Table 1 is a list of the bindings set for each interface widget. The handlers shown in this table are all defined in the source file `Uninstall.applescript`. Notice that some widgets are bound to the same handlers. To identify which widget called the handler, check its name property. For instance, the following code fragment shows how to determine which button was clicked.

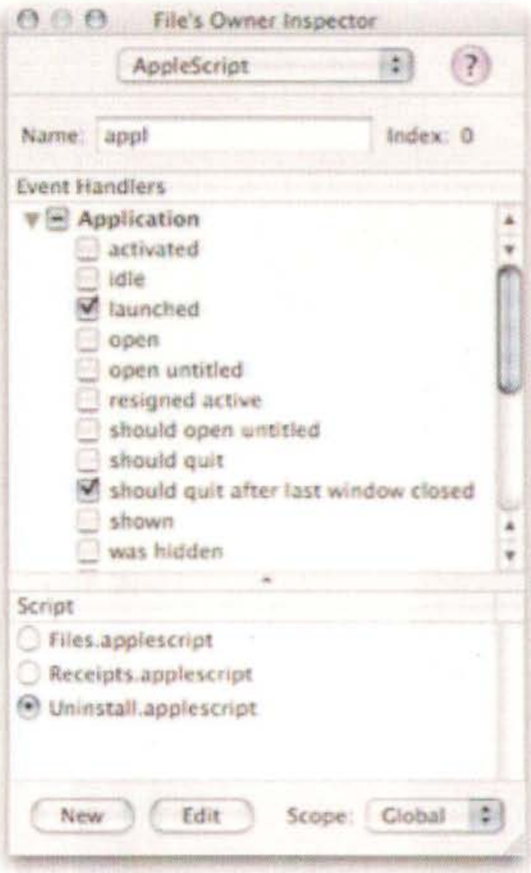
```
on clicked theObject
    local tBtn

    set tBtn to the (name of theObject) as string
    if (tBtn is equal to "prev") then
        -- the Previous button has been clicked
    else if (tBtn is equal to "next") then
        -- the Next button has been clicked
    end if -- (tBtn is equal to "prev")
end clicked -- theObject
```

Widget Name	Class	AppleScript Settings		
		Name	Event	Handler
Uninstall Demo	NSWindow	demo	Nib	awake from nib
Prev	NSButton	prev	Action	clicked
Next	NSButton	next	Action	clicked
Receipts	NSTableView	rcpt	Nib	awake from nib
			Data View	selection changed
Lists	NSTableView	list	Nib	awake from nib
			Data View	selection changed

Table 1. AppleScript settings for the Uninstall UI widgets.

Not shown in the table are the bindings for the application itself. To bind the application, select the **File's Owner** icon on the **MainMenu.nib** window. Then, from the **Show Inspector** dialog, click on the **Application** checkbox. Then set the bindings as shown in Figure 6.


File's Owner Inspector

AppleScript

Name: appl Index: 0

Event Handlers

- [–] Application
  - [ ] activated
  - [ ] idle
  - [✓] launched
  - [ ] open
  - [ ] open untitled
  - [ ] resigned active
  - [ ] should open untitled
  - [ ] should quit
  - [✓] should quit after last window closed
  - [ ] shown
  - [ ] was hidden

Script

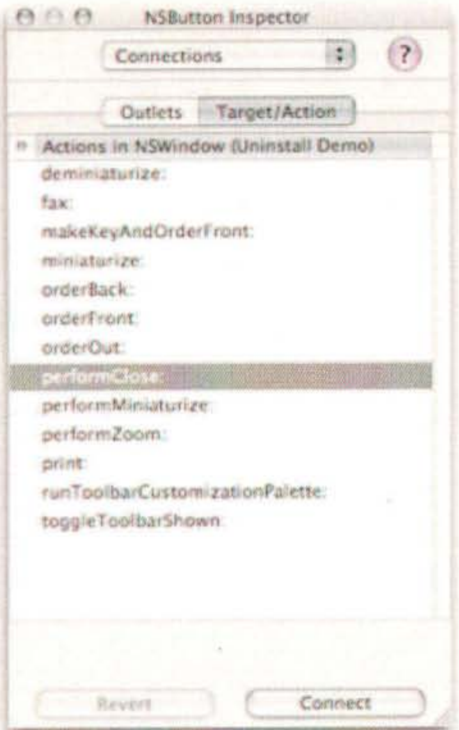
- ○ Files.applescript
- ○ Receipts.applescript
- ● Uninstall.applescript

Figure 6. Binding the AppleScript application.

Binding the Cancel button

On the other hand, the Cancel button is not bound to any AppleScript handler. Instead, it is bound directly to an action handler.

To set the binding, control-drag a line from the button to the main window. This will display the Show Inspector dialog with the Connections panel view active. Click on the Target/Action tab on the dialog. Then choose performClose: from the list of window actions (Figure 7).


NSButton Inspector

Connections

Outlets | Target/Action

Actions in NSWindow (Uninstall Demo)
deminiaturize:
fax:
makeKeyAndOrderFront:
miniaturize:
orderBack:
orderFront:
orderOut:
performClose:
performMiniaturize:
performZoom:
print:
runToolbarCustomizationPalette:
toggleToolbarShown:

Revert | Connect

Figure 7. Selecting the performClose: action.

The Quit Uninstall menu item is also bound in the same way. But this will be left as an exercise to the readers.

Building with Xcode

The Xcode project Uninstall contains three AppleScript source files. The Uninstall.applescript file has all the handlers called by the UI widgets. The Receipts.applescript file has the code to access the Receipts directory. Finally, the Files.applescript file has the code for processing the BOM file. It also has the code that will do the uninstall task.

For reasons of length, this article will only show code that is relevant and interesting. Readers can always view the entire source files by downloading the project from the MacTech website.

Accessing the Receipts directory

Shown in Listing 3 is the function handler that retrieves the contents of the Receipts directory. It takes a path to the directory as its input argument. It returns the results of the retrieval as a list of records.

First, the handler calls the list folder function to read the directory contents. The function responds by returning the contents as a list of filenames. Next, the handler parses each filename in the list. If the name belongs to a receipt bundle, the handler retrieves its bundle signature. Otherwise, the handler proceeds to the next name.

The handler uses the bundle signature to create a record together with the name. When done, it appends the record to the return list `tPkg`.

Long Distance

2.9¢ Per Minute!

Straight 6 second billing increments

Excellent rates on intrastate, intralata/toll calls and international calling with no term contract.

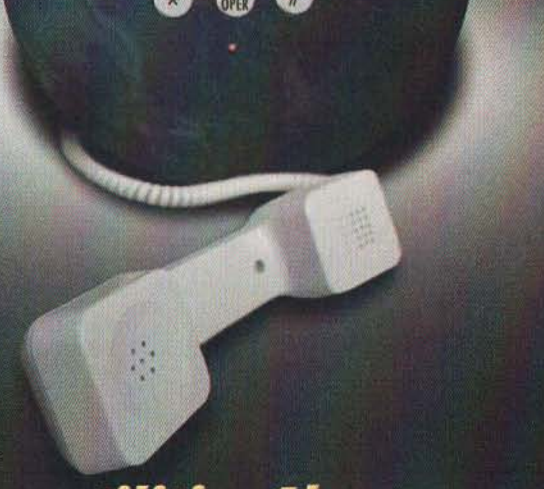
Toll Free (800/888/877/866) service, same low per minute rate for incoming calls.

10 cents per minute calling card.

Detailed billing directly from OPEX.

Quality electronic and telephone customer support.
No monthly billing fee if your bill is over $20.00 each month.

(NOTE: $2.00 billing fee is charged when your bill is under $20.00.)



www.utilities4less.com

Listing 3. Retrieving a list of receipt bundles (Receipts.applescript).

```
property pRcptRec : {bnom:"", bsig:""}

to getBundles from aPath
    local tLst, tPkgs
    local tItem, tNom
    local isPkg

    -- read the contents of the directory
    set tLst to (list folder aPath)

    -- parse the list results
    set tPkgs to {}
    copy pRcptRec to tRec

    repeat with tNom in tLst
        -- prepare a path to a list item
        set tItem to aPath & ":" & tNom
        get info for file tItem

        -- is the item a bundle?
        set isPkg to package folder of result

        if (isPkg) then
            -- retrieve the bundle signature
            set tSig to (getBundleSignature for tItem)

            -- update the record template
            set bnom of tRec to tNom
            set bsig of tRec to tSig

            -- add the updated record to the list
            copy tRec to the end of tPkgs
        end if -- (isPkg)
    end repeat -- with tNom in tLst

    -- return the retrieval results
    return (tPkgs)
end getBundles -- from aPath
```

Reading the BOM file

Listing 5 shows the function handler used to convert the BOM file. It also shows how to use the `do shell script` function to call the `lsbom` tool. The handler takes a path to the receipt bundle as its input argument. If successful, it returns the file path to a temp file; otherwise, it returns an empty string.

First, the handler gets a path to the `TemporaryItems` directory. It converts the path to a POSIX formant, and appends the name of the temp file `bom.out`. This file will store the output results of the lsbom tool.

The handler then prepares the script to be executed with the `do shell script` function. The script consists of the file path to the BOM file, as well as the path to `bom.out`. For example, if the target receipt is `Sample.pkg`, the script will read as follows.

```
    lsbom -p fs
/Library/Receipts/Sample.pkg/Contents/Archive.bom ¬
                    > ~/Library/TemporaryFiles/bom.out
```

Note that, in actual practice, the entire script will consists only of a single line. It will also state the full path to the `bom.out` file on the user home directory.

Notice as well that a `-p fs` option is passed to the lsbom tool. This option tells the tool to display *only the file paths and sizes* of each BOM item. If the item happens to be a directory, its entry in the `bom.out` file *will not have any size data.*

Listing 5. Generating a BOM file for a given receipt (Files.applescript).

```
property pBOMPath : "/Contents/Archive.bom"
property pBOMTemp : "bom.out"
property pCmd : "lsbom "
property pOpts : "-p fs "

to getBOM for aRcpt
    local tTmp, tCmd

    -- retrieve a path to a temp directory
    set tTmp to path to temporary items from user domain
    set tTmp to POSIX path of tTmp

    -- prepare the output file path
    set tTmp to tTmp & "/" & pBOMTemp

    -- prepare the shell command
    set tCmd to pCmd & pOpts
    set tCmd to tCmd & aRcpt & pBOMPath
    set tCmd to tCmd & " > " & tTmp

    -- execute the shell command
    do shell script tCmd

    -- was it successful?
    try
        set tTmp to (POSIX file tTmp) as string
        alias tTmp
        return (tTmp)
    on error tErr number tTyp
        display dialog ("[FATAL]" & tErr as string)
        return ("")
    end try
end getBOM -- for aRcpt
```

Listing 7 shows the function handler used to parse the contents of the `bom.out` file. It takes the path to that file as its input argument. When done, it returns a list of records, each record representing a BOM entry.

The handler first opens a read-only access to the `bom.out` file. It reads all the entries in the file, and then closes the access.

The entries consist of a list of strings. Each entry alternates between the file path and size of a BOM item. The handler creates a BOM record for each entry. It then appends the record to the list variable `tBOM`, which is returned to the calling handler.

Listing 7. Reading the BOM file (Files.applescript).

```
property pBOMRec : {fnom:"", fsiz:""}
property pLF : 10
property pHT : 9

to loadBOMItems from theFile
    local tSrc, tSiz, tPos, tLen
    local tLst, tBOM, tDat, tRec, tNom
    local tTkn, tOdd

    -- initialize the following locals
    set tBOM to {}
    set tTkn to {}
    set tTkn to tTkn & (ASCII character pLF)
    set tTkn to tTkn & (ASCII character pHT)

    try
        -- start a read-only access to the file
        open for access theFile without write permission
        set tSrc to result
        if (tSrc > 0) then
            -- read the contents of the file
            read tSrc using delimiter tTkn
            set tLst to result

            -- close the read-only access to the file
            close access tSrc
        end if -- (tSrc > 0)

        -- parse the BOM entries
        copy pBOMRec to tRec
        set tLen to the length of tLst

        if (tLen > 0) then
            repeat with tPos from 1 to tLen by 2
                -- retrieve the following BOM items
                set tNom to item tPos of tLst
                set tSiz to item (tPos + 1) of tLst
                if (tSiz is equal to "") then
                    set tSiz to "-1"
                end if -- (tSiz is equal to "")

                -- prepare the BOM record
                set fnom of tRec to (tNom as string)
                set fsiz of tRec to (tSiz as string)

                -- append the record to the return list
                copy tRec to the end of tBOM
            end repeat -- with tPos from 1 to tLen by 2

            -- remove the first two items
            set tLen to length of tBOM
            set tBOM to items 3 thru tLen of tBOM
end if -- (tSiz > 0)
    on error tErr
        -- something wrong has happened
        display dialog ("[FATAL] loadBOMItems:" & tErr as string)
    end try

    -- return a list of BOM entries
    return (tBOM)
end loadBOMItems -- theFile
```

Removing a BOM item

There are a many ways to remove a software product. The direct way is to locate the topmost directory from the BOM, and delete it together with its contents. Another way is to select specific items from the BOM for deletion. Choosing the right approach depends on the aim of the uninstaller. The Uninstall project, for instance, uses a variant of the second approach.

Shown in Listing 9 is the function handler that will remove a BOM item. It is called after the user selected an item from the Files listbox, and clicked on the **Uninstall** button.

The handler first creates an **Uninstall** folder in the `TemporaryItems` directory. Next, it moves the BOM item from its original path to the folder. It also deletes the item at the specified path. Once the handler completes its task, it returns a `true` to the calling handler. Otherwise, it returns a `false` if any errors occurred.

Listing 9. Removing a BOM item (`Files.applescript`).

```
    property pDirTrash : "Uninstall"

    to removeTheItem given path:aTgt, folder:aDir
        local tTmp, tBin, tCmd

        -- initialize the following locals
        set tTmp to path to temporary items from user domain
as string
        set tBin to tTmp & pDirTrash
```

macforge.net™

MacForge indexes and tracks open source projects that run on the Mac, or are likely to without modification. Thanks to MacForge, there's no need to sift through huge listings of open source that you can't use. With categories, filters, and more, MacForge makes it easy to find what you need.

MacForge:

Your Gateway to Mac Open Source

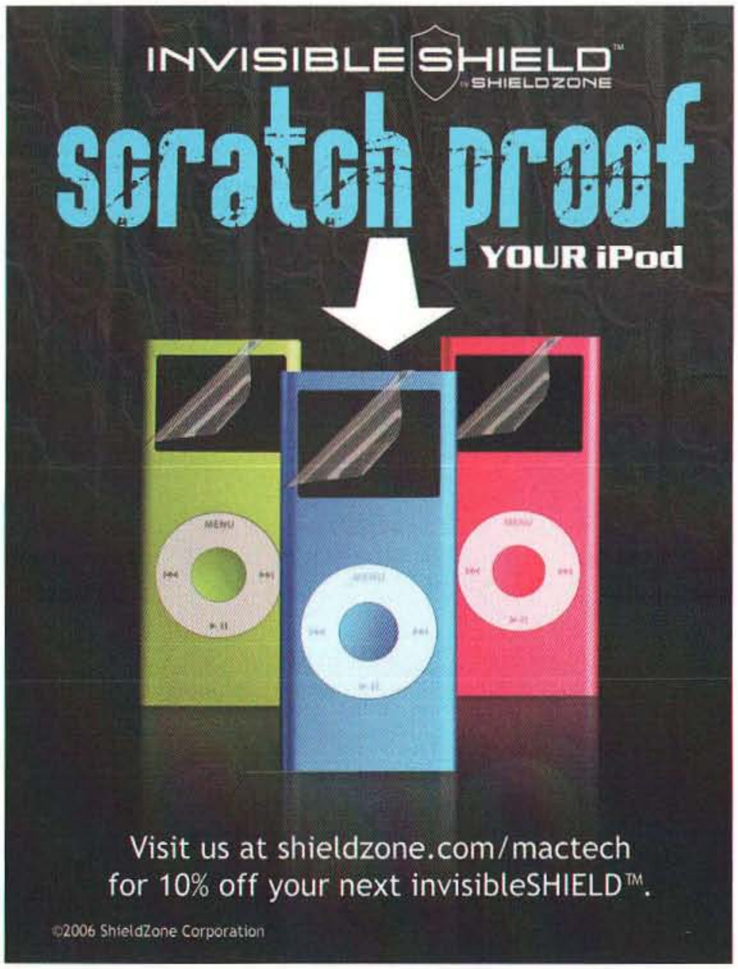
www.macforge.net

Sponsored by MACTECH

```
        -- create the temporary uninstall directory
        tell application "Finder"
            try
                if not (exists alias tBin) then
                    make new folder at folder tTmp ¬
                        with properties {name:"Uninstall"}
                end if -- (exists alias tBin)
            on error tErr
                display dialog ¬
                    "[FATAL] Failed to create the Uninstall
directory"
                return false
            end try
        end tell -- application "Finder"

        -- attempt to remove the item
        tell application "Finder"
            try
                -- is the item a directory or a file?
                if (aDir) then -- it is a directory
                    -- test delete the directory
                    move folder aTgt to folder tBin with
replacing
                    delete folder aTgt
                else -- it is a file
                    -- test delete the file
                    move file aTgt to folder tBin with
replacing
                    delete file aTgt
                end if -- (aDir)
            on error tErr
                display dialog "[ERROR] " & tErr
                return false
            end try
        end tell -- application "Finder"

        -- the removal was successful
        return (true)
    end removeTheItem -- given path:aTgt, type:aTyp
```



Visit us at shieldzone.com/mactech for 10% off your next invisibleSHIELD™.

Notice that the handler uses the Finder to delete the BOM item. While this works in most cases, it will fail if the item is inside a restricted directory such as `/usr`. For that case, replace the delete code with the following script statements.

```
    set aTgt to the POSIX path of alias aTgt
    set tCmd to "rm -Rf " & aTgt
    do shell script tCmd with administrator privileges
```

The above statements will prompt the user to validate the deletion that is about to occur.

Also, notice that the handler first makes a copy of the BOM item it is about to delete. This gives the user a chance to restore the deleted item back to its former location.

Final Thoughts

Product removal is just as important as product installation. Though most products are easy to remove manually, some require the use of an uninstaller tool. The tool will peruse the receipt package for the product, and delete all the files that belong to that product. This will help ensure that future products installations will be more successful.

AppleScript Studio makes it quite easy to build an uninstaller tool. It has a much lower learning curve compared to Cocoa. This alone makes for a faster build and deployment cycle. It allows the addition of a user-friendly interface, which is not possible through shell scripts.

Hopefully, this article helps to get you started in writing your own uninstaller. Until Apple adds an uninstall option to the Installer tool, writing your own is, for now, the next best solution.

Bibliography and References

Apple Computers. *AppleScript Resources.* Retrieved 2007 Feb 24. Online:
http://www.apple.com/applescript/resources.

Apple Computers. "do shell script in AppleScript". *Technical Note TN2065.* Copyright 2003, 2005, 2006. Apple Computers, Inc.

Apple Computers. "Administrative Accounts". *An Introduction to MacOS X Security for Web Developers.* Copyright 2007. Apple Computers, Inc. 2004 Aug 25. Online:

http://developer.apple.com/internet/security/securityintro.html

Apple Computers. "lsbom – list contents of a bom file". *Mac OS X Man Pages.* Copyright 2003. Apple Computers, Inc. 2003 Apr 16. Online:

http://developer.apple.com/documentation/Darwin/Reference/Manpages/
man8/lsbom.8.html.

MT

About The Author

JC is a freelance engineering writer currently residing in North Vancouver, British Columbia. He divides his time between writing technical articles, and teaching origami at his local district's public library. He can be reached at anarakisware@icmail.net.

by Michael R. Harvey

AMCC 3ware Sidecar Kit

When it's time to step up to more and better

Recently, we reviewed a small NAS unit that was good for smaller home offices or work groups. That product, however, lacked features critical to any group larger than a few people or workstations. The AMCC 3ware Sidecar is the next step up. It is a full hardware RAID storage device, but not so big as to be cost prohibitive for smaller installations. This desktop RAID unit is an ideal starting point as your storage needs grow beyond a single disk, and a CD backup.

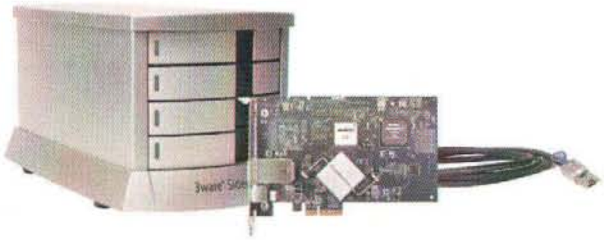
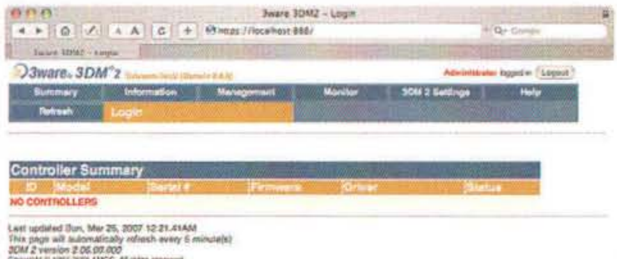


Figure 1: The AMCC 3ware Sidecar Kit

The unit itself is a four bay SATA enclosure. It attaches to any G5 or newer Mac with PCI Express, via a multilane SATA II cable to an AMCC 9590E-4ME controller card that is included in the kit. The card itself is 4 lanes, and so will work in either a 4x or 8x PCI Express slot. There also is a controller cable that runs between the RAID unit and the card providing device management and information to the set up and monitoring software. The unit can be placed anywhere with good airflow. It is small enough to fit on top of a G5/Mac Pro tower.

The software itself is two parts. First, a background process that runs at boot and does the monitoring, and, second, a web based interface for configuration and reporting. The system requirements are OS X 10.4.6 (10.4.8 for Intel based Mac Pros), and Java 1.5 or later. The installer is dreadfully slow, but it's Java based, so that behavior is expected. The web interface, however, is responsive, and includes a good help system.

Hardware and software installation are pretty straightforward. Put the parts where they belong, link up the cables, make sure the RAID unit is powered on before the computer, and then start up. To configure the Sidecar, you type into a web browser: https://localhost:888. From that web-based interface, after logging in as an administrator, you can select drives and choose the RAID level you want. We suggest either a 3 disk RAID 5 with a hot spare (you can designate a drive as a spare in the web management software), or a RAID 10. Both these configurations require all four drive bays be used, but they give you either very good performance and fault tolerance (RAID 10) or high capacity with hot fail over in case of a drive problem, along with a performance boost (RAID 5). After the 3ware interface has your unit set up, you'll need to partition it using Disk Utility. You are good to go after that.



Controller Summary

ID	Model	Serial #	Firmware	Driver	Status
NO CONTROLLERS					

Last updated Sun, Mar 25, 2007 12:21.41AM
This page will automatically refresh every 5 minute(s)
3DM 2 version 2.06.00.000
Copyright © 1997-2006 AMCC. All rights reserved

Figure 2: The web control interface, displaying an error condition

The one obvious failing of this unit compared to the NAS we reviewed earlier is availability over the network. Two options here: In smaller settings, you can use Hornware's SharePoints to enable sharing using OS X. Your other option is to attach the unit to an Xserve, or othe Mac running OS X Server, and share it out that way. It's not an XServe RAID, but then again, that much hardware is not what you are looking for in the scenario we are assuming for this review.

Also, at NAB 2007, AMCC announced that support for Windows systems was now available for the 3ware Sidecar. This opens up more options if your backend is running on Windows Server. You can still get RAID level storage made available to your Mac desktops that way.

The base unit, that does not include disk drives, costs $1295, and can be purchased directly from AMCC, as well as a few resellers. This allows you to add whatever size drives you need to complete the unit.

The AMCC unit is a very good starting point for moving into RAID quality storage. The price is not unreasonable, and the options the unit provides, as well as the reporting capabilities, make this product an excellent choice for moving from basic storage to fault tolerant, and better performing, RAID storage.
www.3ware.com

MT

continued from p.80

In terms of changing Apple itself, without question I'd like to change their culture of secrecy. Users aren't terribly affected by this, they just get surprised when Apple unveils a major new product like the iPod. However, as far as being a developer on the platform, getting information and details on anything from Apple is like pulling teeth. From bug reporting to specifications of existing and upcoming products, we've always had a hard time getting to the people we need at Apple through what few established channels exist. This is sort of accepted by the Mac development community, but Apple's complete lack of communication with its third-party developers is far from the norm with most companies.

What's the coolest tech thing you've done using OS X?

I'll use this to talk about something we did at Rogue Amoeba. When working on Airfoil 2, back at the end of 2005, we decided to attempt to add support for sending audio to multiple AirPort Express units. And while it is straightforward to send audio to multiple units, keeping the playback between them properly synchronized, is not. Quentin worked a great deal trying to find a way to synchronize audio with the limited options provided by the AirTunes protocol. After about two months of research, a way was found by reading undocumented quality-of-service reports from the Airport Express, combined with an audio stretching/shrinking algorithm. It worked, but was eventually fall out of sync. At this point Mike Ash took over, and spent another month working on some high-level math to get units working nearly perfectly in sync with one another.

We released this January 7th, with the hopes of revolutionizing the way people used AirPort Express units. On January 10th, Apple released iTunes 6.0.2 with AirTunes 2, which quietly added support for multiple synchronized units as well. To solve the synchronization problem, they completely redid the AirTunes protocol (called AirTunes 2), to support synchronization natively.

We were eventually told by Apple, that they didn't believe synchronization with AirTunes 1 was possible, at all. So we had achieved the impossible, but no one ever knew.

Where can we see a sample of your work?

The software I've worked with everyone at Rogue Amoeba to create, is available on our website at http://www.rogueamoeba.com/. Free trials of all our software can be downloaded there, and license keys can be purchased through our online store.

The next way I'm going to impact IT/OS X/the Mac universe is:

I think Rogue Amoeba's next product, as yet unannounced, will really have a major impact. Audio Hijack and Audio Hijack Pro have focused on recording in general. With our next application, we're going back to Audio Hijack's roots and doing radio recording right. If you've ever watched and record TV with Tivo, you'll have some idea of what we're doing. That'll be coming in the first quarter of 2007, and we're really excited.

Anything else we should know?

I hope people will check out our software, we've got free trials of all of it, as well as several useful Freebies. If that's not interesting enough, we've also got stylish t-shirts and plush toys of our mascot Ammo. http://www.rogueamoeba.com/merchandise/

MT

Is your computer running slow?

Maybe it needs a boost.

Make your life a little faster and easier,

get better RAM.

betterRAM.com sm

Name brand quality. Value pricing. Lifetime guarantee. World class support.

Toll Free: (800) 895-3493 • Outside US/Canada: 805-494-9797 • Fax: 805-494-9798

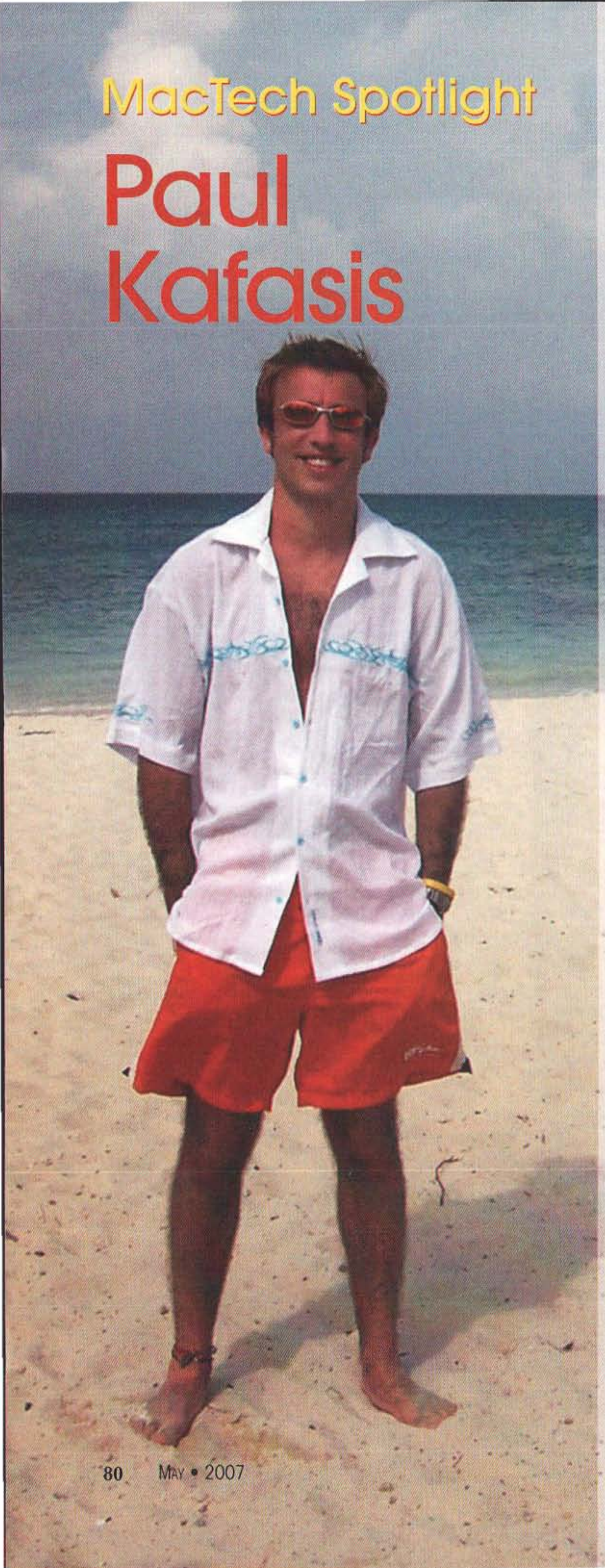
www.betterram.com

Advertiser/Product Index

The index on this page is provided as a service to our readers. The publisher does not assume any liability for errors or omissions.

MacTech Spotlight

Paul Kafasis



Your Company?

Rogue Amoeba Software

What do you do?

My official title is CEO/Lackey, which makes me the high and low man on the corporate ladder here. Essentially, it's my job to make sure the programmers can keep on programming. That means I handle product management, business development, marketing, website content, some support, and much more.

How long have you been doing what you do?

I've been working for Rogue Amoeba since we founded it in 2002. I've been handling various aspects of software development (at other companies) for about 8 years total, but Rogue Amoeba has been by far the most successful.

Your first computer:

My first computer was a Mac Plus, in about 1987, with 8 blistering MHz of CPU power.

Are you Mac-only, or a multi-platform person?

As far as my personal use, I'm exclusively on the Mac, and have been since that Mac Plus. I have various levels of understanding of Unix, Linux, Solaris and Windows, from working support jobs in college, and Rogue Amoeba has one Windows port (of Airfoil) that I use a cheap Dell to test. That Dell spends 95% of the day turned off, secretly wishing it was a Mac.

What attracts you to working on the Mac?

I never sat down and said "I'm going to develop software, and I'm going to do it on the Mac". It happened organically - everyone at Rogue Amoeba uses a Mac, so naturally we develop for the Mac.

There certainly are things that make it exciting and keep us here on the Mac, however. First and foremost, the user community is great in terms of purchasing and supporting third party software. People often assume if we developed for Windows, we'd have much higher sales volume. However, it seems that far fewer Windows users purchase third party software, and there are often many alternatives and competitors. The user base may be much larger, but that doesn't instantly mean success.

As well, OS X is a great platform to develop on. It's young, it's fresh, and it's constantly being updated with new technologies. When we first started developing on it (in 10.0), it was a wide open playing field - no one wanted to run Classic if they could avoid it, so old applications could be remade and reworked, and new ideas could be explored.

What's the coolest thing about the Mac?

I hear this question a lot, and I don't think there's one specific thing. It's not "the Dock" or "great third party software". For me, the best part about the whole Apple experience is the attention to detail.

My MacBook's MagSafe connector is a perfect example of this - I've personally pulled at least one laptop off a table by tripping over the power cord, but never again. The accessible case of the G3/G4 towers, the scrolling track pad of newer laptops, the built-in iSight - they all represent an attention to detail and the way users work that I just don't see evident in other machines.

If I could change one thing about Apple/OS X, I'd:

This question is a setup for disaster and retaliation, isn't it? As far as OS X goes, I'd like to see a new/rewritten Finder.

continued on page 78

ALWAYS BY YOUR SIDE.

Small Dog Electronics is the third largest Apple Specialist in the US. We offer over 4000 Mac-compatible products, including new and Apple Refurbished Xserves, Xserve RAID's, RAM, hard drives, networking gear, and much more.

We've earned our place in the Apple community with our superior customer service, great prices, socially responsible mission, and by staying true to the Apple platform.

We are here to help you. Give us a call, or visit us at Smalldog.com!

Small Dog Electronics is 100% Mac-based. We're powered by:
14 Xserves + 4 G4 Servers + 3 T1 lines + Heaps of Cisco Routers + 4 Miles of CAT 5 + 13 kW Propane Back-up Generator + 13 Network Cameras + 1 Firewall + 50 Mac Workstations + 41 Apple Product Professionals + 14 dogs of all shapes and sizes!

Call for a free custom quote for home or business!

Small Dog Electronics

Always by your side.

$6.00 off shipping on your next order!
Simply enter coupon code "mactech" when checking out at Smalldog.com!
Coupon good through 7/31/07

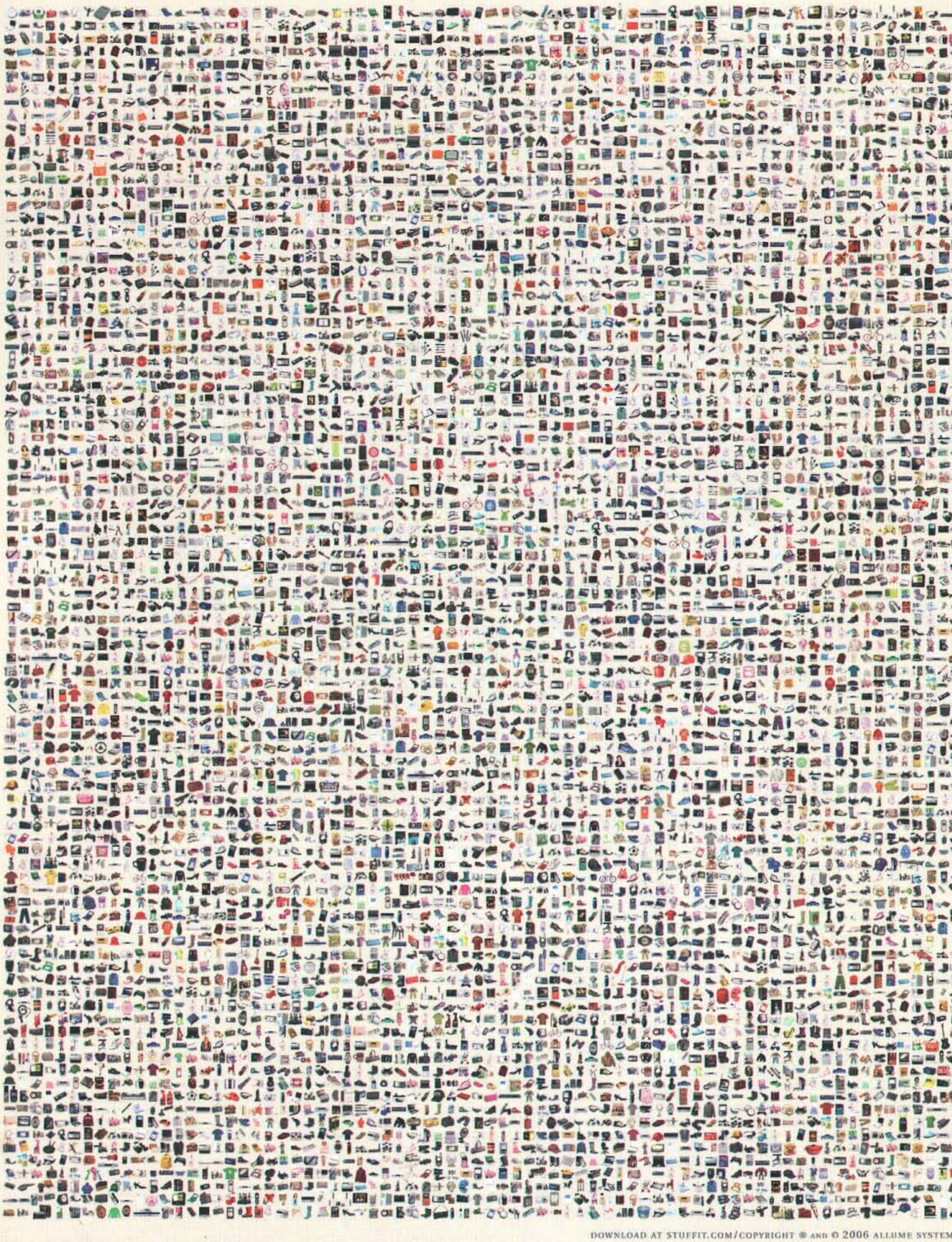
www.smalldog.com 800-511-MACS Apple Specialist

THINGS MADE IN CHINA

COMPRESSED BY

STUFFIT DELUXE®

SOFTWARE FOR MAC/PC



DOWNLOAD AT STUFFIT.COM/COPYRIGHT ® AND © 2006 ALLUME SYSTEMS, INC.